ENHANCED ENERGY AWARE TINYDDS PUBLISH/SUBSCRIBE APPROACH (E-EATDDS)

BY

AWADH MOQBEL GAAMEL

A Thesis Presented to the

DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the

Requirements for the Degree of

MASTER OF SCIENCE

In

COMPUTER NETWORKS

MAY 2017

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN- 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **Awadh Moqbel Gaamel** under the direction of his thesis advisor and approved by his thesis committee, has been presented and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN COMPUTER NETWORKS**.

Prof. Tarek R. Sheltami
(Advisor)

Dr. Ahmad Al-Mulhem
Department Chairman

Dr. Talal M. Al-Kharoubi
(Member)

Dr. Salam A. Zummo
Dean of Graduate Studies

Dr. Anas A. Al-Roubaiey
(Member)

23/5/17

Date

To my parents

ACKNOWLEDGMENTS

First and foremost, I am grateful to Allah for the good health and wellbeing that were necessary to complete this thesis. I would like to express my sincere thanks to my advisor Professor Tarek Rahil Sheltami for his continuous support and guidance of my master study and related research. Also, I would like to thank him for his insightful comments, patience and motivation. Besides my advisor, I would like to thank the rest of my thesis committee: Dr. Talal Al-kharobi and Dr. Anas Al-Roubaiey for their comments, suggestions and encouragement, but also for the different questions which incented me to widen my research from various perspectives.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

BLTDDS	:	Broker Less TinyDDS
DDS	:	Data Distribution Service
DefTDDS	:	Default TinyDDS
EATDDS	:	Energy-Aware TinyDDS
E-EATDDS	:	Enhanced EATDDS
EED	:	End-to-End Delay
HyTDDS	:	Hybrid TinyDDS
SW	:	Software
HW	:	Hardware
OMG		Object Management Group
GUI	:	Geographical User Interface
MCU	:	Micro-Controller Unit
PDR	:	Packet Delivery Ratio
Pub/Sub	:	Publisher/Subscriber
QoS	:	Quality of Service
TOSSIM	:	TinyOS SIMulator

JTOSSIM	:	Java TOSSIM
WSAN	:	Wireless Sensor/Actuator Network
WSN	:	Wireless Sensor Network
OEM :	:	Online Energy Model

ABSTRACT

Full Name : [Awadh Moqbel Nasser Gaamel]

Thesis Title : [Enhanced Energy Aware TinyDDS Publish/Subscribe Approach (E-EATDDS)]

Major Field : [Computer Networks]

Date of Degree : [May, 2017]

Over the past few years, the cost of electronic devices and sensor networks decreased rapidly enforcing almost all users' requirements to use those devices. These devices provide low cost processing over wireless sensor networks (WSNs)/ wireless sensor and actuator networks (WSANs). The task of data management in WSNs is a vital issue that can be performed with limited resources such as processing, memory and energy. Data Distribution Service (DDS) is a prominent standard used in the industry and academia communities to support real-time distributed systems depending on publish/subscribe (pub/sub) model. TinyDDS is a light-weight middleware which is a partial porting of the DDS and implemented over TinyOS code. The original version of TinyDDS is called the Default TinyDDS (DefTDDS). Broker-Less TinyDDS (BLTDDS) and Hybrid TinyDDS (HyTDDS) are proposed protocols that added several improvements to the DefTDDS. Energy Aware TinyDDS (EATDDS) protocol is proposed to deal directly with the energy consumption metric. In this thesis, we conduct a comparative study between DefTDDS, BLTDDS and HyTDDS in terms of throughput, PDR, End-to-End delay and energy consumption. Moreover, we propose a new approach titled: Enhanced Energy Aware TinyDDS (E-EATDDS) by enhancing the EATDDS protocol in order to improve the energy consumption. We use TinyOS simulator in our implementation and evaluation. The results showed that E-EATDDS outperform BLTDDS, HyTDDS and EATDDS in terms of energy consumption and packets sent per joule.

ملخص الرسالة

الاسم الكامل: عوض مقبل ناصر جعامل

عنوان الرسالة: تطوير برنامج النشر والإشتراك للتحكم في معلومات الطاقة لشبكات الإستشعار اللاسلكية

التخصص: شبكات الحاسب الآلي

تاريخ الدرجة العلمية: مايو، 2017

على مدى السنوات القليلة الماضية، انخفضت تكلفة الأجهزة الإلكترونية وشبكات التحكم اللاسلكية (WSAN) عن طريق توفير متطلبات المستخدم لتلك الأجهزة ذات التكلفة المنخفضة من حيث معالجة شبكات الاستشعار والتحكم اللاسلكية. إن مهمة إدارة البيانات في شبكات الاستشعار اللاسلكية مسألة مهمة جدا والتي يمكن توفيرها باستخدام موارد محدودة مثل المعالجة والتخزين والطاقة. خدمة توزيع البيانات (DDS) هو معيار بارز يستخدم في المجالات الصناعية والأكاديمية لدعم الأنظمة الموزعة ذات الوقت الحقيقي اعتمادا على نموذج النشر والإشتراك (نشر/إشتراك pub/sub). TinyDDS هي نسخة مصغرة ومطورة من DDS والتي صممت لتلبي احتياجات الشبكات الاستشعارية. النسخة الإفتراضية او الاصلية من TinyDDS تسمى Default TinyDDS/DefTDDS. وإعتمادا على النسخة الاصلية من TinyDDS فقد تم تطوير Broker-Less TinyDDS/BLTDDS و Hybrid TinyDDS/HyTDDS كحلول بديلة يمكن استخدامها للتغلب على بعض المشاكل الموجودة في النسخة الأصلية والتي تسببها مشكلة عنق الزجاجة. EATDDS هو حل آخر تم تطويره لكي يقلل من الطاقة المستهلكة في الشبكات اللاسلكية ويزيد من عمر الشبكة. هذا العمل يقدم مقارنة شاملة بين DefTDDS، BLTDDS و HyTDDS من حيث الإنتاجية، نسبة تسليم الرزم، زمن التأخير بين المرسل والمستلم و الطاقة المستهلكة. وأيضا، نقترح بروتوكول E-EATDDS والذي تم تطويره لتحسين أداء EATDDS من حيث استهلاك الطاقة. لتنفيذ وتقييم هذه البروتوكولات، تم إستخدام برنامج TOSSIM كأداة محاكاة. النتائج توضح ان E-EATDDS يستطيع ان يتفوق على BLTDDS، HyTDDS و EATDDS من حيث استهلاك الطاقة وعدد الحزم المرسلة في كل جول.

CHAPTER 1

INTRODUCTION

Wireless sensor network (WSN) is a collection of deployed embedded nodes, each node has the ability of sensing, processing and communicating. Sensor networks potentially comprise of tens to hundreds of tiny devices with limited energy and individual resources. They are usually deployed to an area of interest to mainly monitor the state of physical or environmental conditions such as sound, temperature, pressure, vibration or motion. Traditionally, the deployed sensors gather and convey the data to the sink node by using one-to-many communication pattern [1, 2]. The sink node acts as an interface between the network and users as presented in Figure 1. Hence, the main task on WSN is to identify and gather the data out of the enclosing area without any noticeable encounter. Applications of WSN include but not limited to Health care monitoring, Environmental/Earth sensing, Industrial monitoring, Air pollution monitoring and Water quality monitoring benefit from WSN functionality.

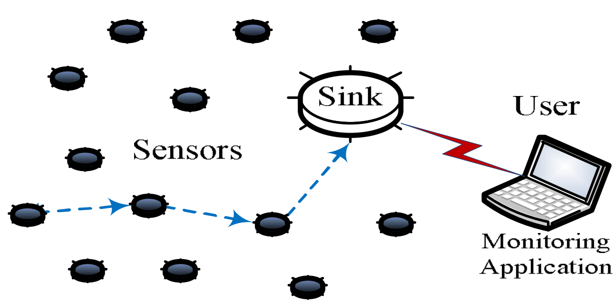

Figure 1. Architecture of Traditional WSN

Nonetheless, owning to recent technological advances in the field of sensor-based network technology, the Wireless Sensor and Actor Networks (WSANs) appear to be a suitable technology purposely for making decisions within network, in which the network identifies and responds without visiting external and control applications [3]. In WSAN applications, the process automation can be grouped into fully and partial automated applications per the data exchange [4] is illustrated in Figure 2. The interaction of the partial automation in which the sink is considered in making decision, which is more unified and organized. However, it induces more lag.

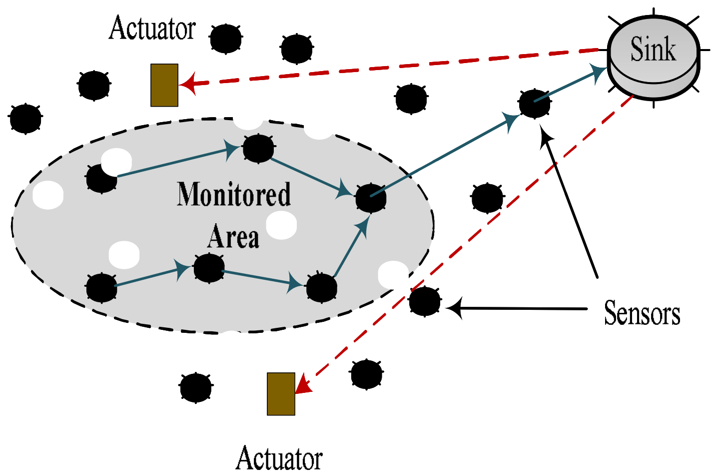

Figure 2. The Partial Automation Interaction in WSAN

Figure 3 presents the interaction of the fully automation process where sensors sense and then send data directly for processing and reacting by the actuators that respond to the local data analysis result. For practical applications, the fully automated approach is considered more appropriate than the partial automated approach in terms of the time and overhead.

Following the definition of WSAN, we need to define the publish/subscribe (pub/sub) model and its aptness for WSAN. As a messaging based communication model, the pub/sub paradigm serves as a medium where data is delivered to a logical data space, referred to as middleware by senders.

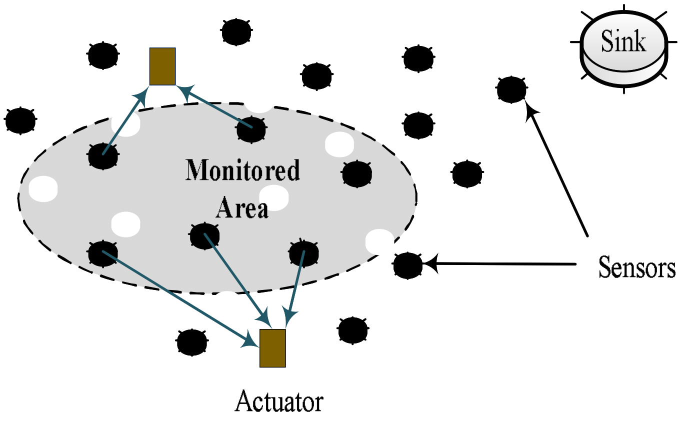

Figure 3. The Partial Automation Interaction in WSAN

The senders, called publishers, do not know who or where the receivers, referred to as subscribers are. Likewise, only the interested data is received by the subscribers with no idea where or who the publishers are. The pub/ sub interface was created to handle large-scale distributed practical applications. A suitability analysis was done for pub/sub model and the following remarks were made [5].

1. Where the setup is very large and the transmission of data is distributed between many users (usually in sensor networks), pub/sub model becomes very rewarding.

2. Where the updating of events or data becomes seldom, pub/sub model tends to be very appropriate.

3. In addition, where the degree of collective interest is of importance, pub/sub model becomes the right choice. For instance, the data gathered by sensors in WSAN applications highly has a similar interest as the actuators, several sinks or applications.

4. In less user involvement applications, pub/sub is more superior to request/replay model. (5) Data upgrades are swiftly transferred to subscribers in pub/sub model. Hence, more suitable in practical applications where time is of essence. For instance, in battlefield surveillance

WSN.

5. Pub/sub becomes less suitable in cases where published data is rarely used by clients.

In order to achieve scalability and flexibility in pub/sub service, decoupling between the subscribers and publishers is presented in three dimensions known as decoupling properties.[6].

a. ***Time dimension****:* the participating parties do not necessarily have to be part of the communication at the same time. For instance, an event can be published by a publisher whiles the subscriber to receive that event may come later even beyond the existence of the publisher; similarly, a subscriber may subscribe to a particular event which is yet to be published as given in Figure 4. Thus, it is very useful where the nodes interruption rate is high as in the case of high dynamic networks.

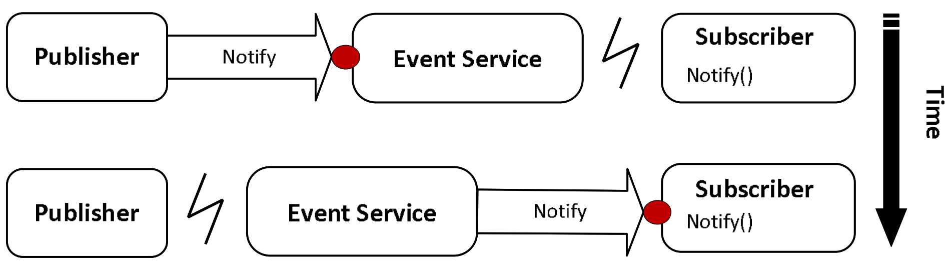

Figure 4. Time Dimension Decoupling

b. ***Space dimension:*** since the main concern is in the event no matter its origin, the publishers and subscribers (the interacting parties) do not need to know each other. The pub/sub services appear to be the mediator between the interacting parties where events are published by a publisher by means of the pub/sub service and delivered to the subscriber from the pub/sub service indirectly as shown in Figure 5.

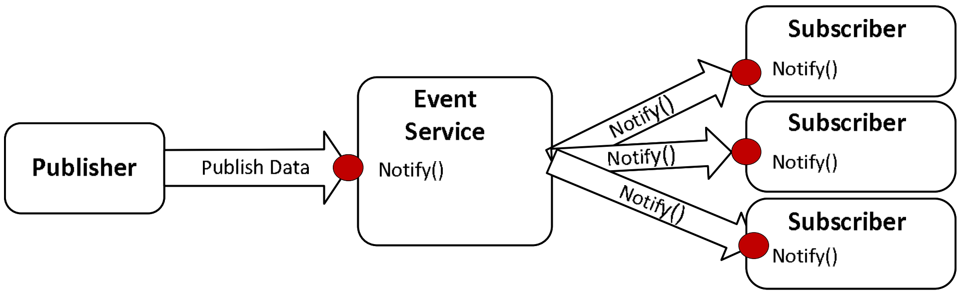

Figure 5. Space Dimension Decoupling

c. ***Synchronization dimension:*** this is to say, there is no blockage on both sides of the subscriber and publisher; in the cause of concurrent tasks execution, there is no blockage between the publishers and subscribers while publishing or subscribing to events as depicted in Figure 6. On the other hand, there is blocking in the end node of synchronous communication models until the other nodes take delivery of the message, which results in static and rigid applications.

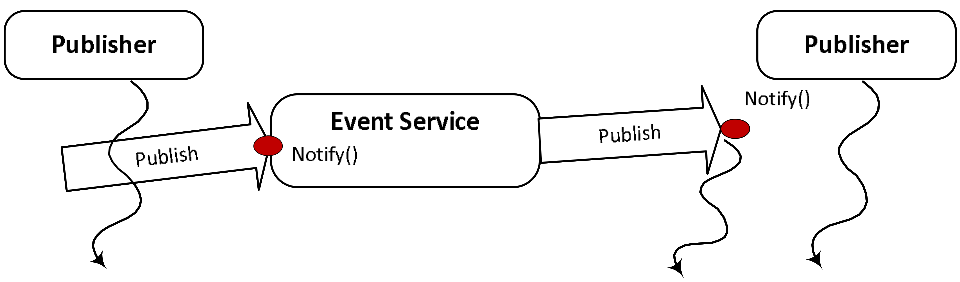

Figure 6. Synchronization Dimension Decoupling

The pub/ sub model has been recognized largely as the most promising application-level communication model for information systems integration [7, 8]. The decoupling characteristics of the model discussed before makes the model more scalable and robust [9]. As a result of these properties, the model becomes the right choice as far as data-centric sensor network applications are concerned. Furthermore, due to the distinct characteristics of the sensor network applications, the pub/sub middleware becomes the suitable solution for such environment [10].

1.1 Background

In this section, we overview the pub/sub communication system and the prominent QoS supported by such system. Also, we describe the specification of DDS standard.

1.1.1 The Publish/Subscribe Communication System in WSANs

As noticed from the previous section, the pub/sub model is designed for wide-range disseminated systems to render them scalable, faster and flexible. Figure 7 shows a simple model depiction of pub/sub system together with its major elements. The main component being the notification service also known as the pub/sub service which mainly stores and controls the subscriptions. As shown in the Figure 7, data in actual applications is spread over the brokers and/or end-node of the centralized system. The mediator between the subscribers and the publishers is mainly the notification service. For a given interest in a particular event, this interest can be expressed by utilizing the subscription function sub (Event), and afterwards, this subscription is matched by the notification service with the current events published by the publishers and conveys to the subscriber, the matched event. In the pub/sub systems, three major operations are utilized: (1) pub (Event) which publishes the events, (2) sub (Event) which subscribes to a particular event, and (3) unsub (Event) which unsubscribes to an event. Participants can be a subscriber or publisher or both simultaneously by pub/sub entity.

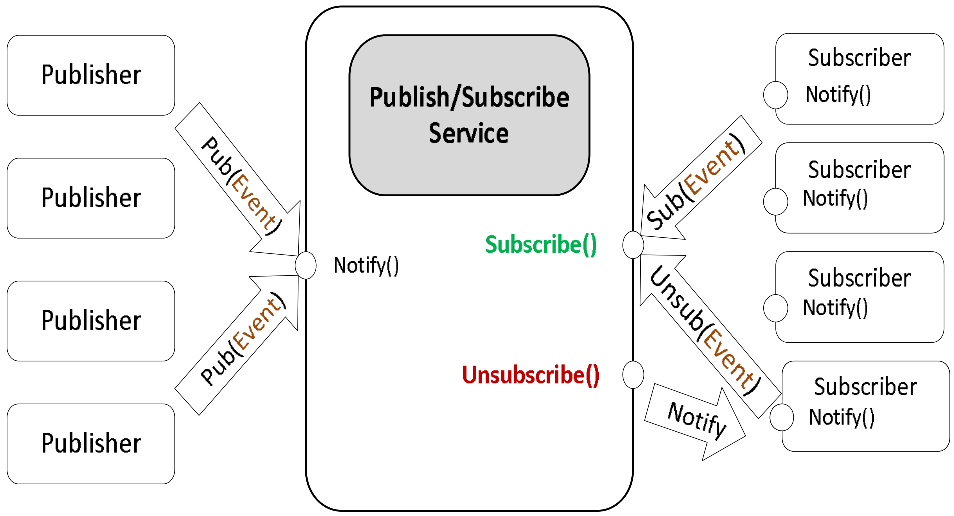

Figure 7. Simple publish/subscribe Communication Model Architecture

1.1.2 QoS in Publish/Subscribe Model

Additionally, to offer QoS support in networks with constrained resource is more challenging problem which is still a subject for research [11] in sensor networks. QoS is expressed by aggregation delay, data accuracy, system lifetime and coverage, in the application layer. In the network layer, QoS can be expressed by throughput, latency, message delay, bandwidth utilization, jitter and loss. In situations, where the requirements of QoS in the application layer could not be fulfilled by the application network layer, the application middleware should bargain in order to get new QoS guarantee between the two layers [12]. Described below are the popular QoSs offered by the pub/sub paradigm.

Priority

This describes a method where data flows are assigned several levels of importance. By doing so, the more important data is processed first by the system. Usually, in WSAN systems, several levels of importance are linked with exchanged messages between the nodes. As an example, usually, failure or attack on notification events is not of the same importance as monitoring readings.

Reliability

This specifies the application network ability to make sure there is reliable data transmission among nodes. In order to ensure that important measurements or notifications produced by a system get to their preferred destination, information contained in pub/sub network application system ought to be transferred in a reliable manner.

Energy-awareness

Battery is usually the main source of energy for WSAN devices. In most situations, it is usually very challenging to replace these batteries. Mostly, the energy is expended in the wireless transmission, in that the energy expended via sensing and computation is relatively small as compared with transmission. As a result, there is the need to sensibly manage the transmission in order to reduce energy utilization so as to maximize the life time of the network. Hence, SA (Sensor/Actuator) devices duty cycles handling become a serious issue. Whenever there are no new data to process, the WSAN devices ought to be in sleep mode and it will wake up and publish when there are fresh data. The sleeping time can be varying from seconds to several hours in most implementations. To save energy, the middleware methods ought to be aware of this. Nevertheless, QoS support and energy efficiency happen to be two differing requirements and as such, the WSAN design ought to efficiently draw the balance among them [13].

Deadline

Also, referred to as maximum allowed latency, deadline describes the latest time by which the subscriber would wait for an update. The data in real-time systems would be dropped if it is acquired beyond a particular threshold since it would make no sense. This is usually used in scheduling the transmission of event by using the earliest deadline first (EDF) algorithm.

1.1.3 OMG DDS Standard Specification

Data Distribution Service (DDS) specification has been standardized by Object Management Group (OMG) to provide standard interfaces used for pub/sub middleware. These interfaces shown in Table 1 are used for event publication and subscription over Interface Definition Language (IDL). DDS standard has two main layers namely, Data-Centric Publish-Subscribe (DCPS) layer and Data Local Reconstruction Layer (DLRL).

Table 1. DDS Interfaces

Interface	Functions
Entity	enable(); set_qos(); get_qos(); set_listener(); & get_listener();
DomainParticipant	create_publisher(); create_subscriber(); create_topic() & create_contentfilteredtopic();
Data	
Topic	
DomainEntity	
QosPolicy	name: string
Publisher	create_datawriter()
Subscriber	create_datareader()
DataWriter	write()
ContentFilteredTopic	get_expression_parameters(); set_expression_parameters() filter_expression : string
DataReade	read(); on_data_available()
SubscriberListener	on_data_on_readers()

The first layer (DCPS) is a low-level fundamental layer that provides a collection of interfaces used for event publication and subscription. The interfaces define each event with an associated topic and help applications to advertise their intents of becoming subscribers and publishers. The second layer (DLRL) is a high-level optional layer that allows the subscribers to access an event as if it was locally available and get events from remote publishers. Figure 8 shows the architecture and main components of DDS standard.

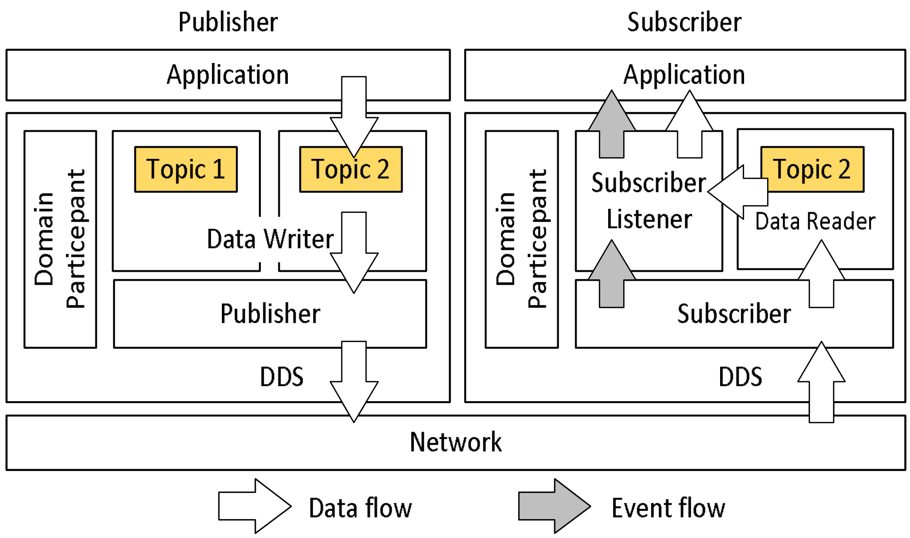

Figure 8. The Architecture of DDS

Figure 9 and Figure 10 present how the publication and subscription processes can be performed by using the DDS components, respectively. For each domain, each node represents an instance of the DomainPartipant component. A domain is a context that determine to which a DDS application can be associated. References are maintained by the DomainPartipant component to all objects that have the same domain.

As shown in Figure 9, when a publisher's application creates an instance of the Publisher components along with the local DomainParticipant, it instantiates Topic according to the generated event(s). The application can instantiate DataWriter for each topic. The DataWriter component acts as an access point to write out the event data. In the same way, when a subscriber (as in Figure 10) needs to subscribe to an event, it has to instantiate the subscriber component with the local DomainParticipant.

Afterward, it can create a Topic instance based on the event(s) which a subscriber is interested in. Each topic is able to identify the content type of an event. Also, the application creates an instance of DataReader and SubscriberListener for each topic to be used as access points that can read events in the future. The subscription of an event is sent toward a publisher application(s) by the Subscriber component.

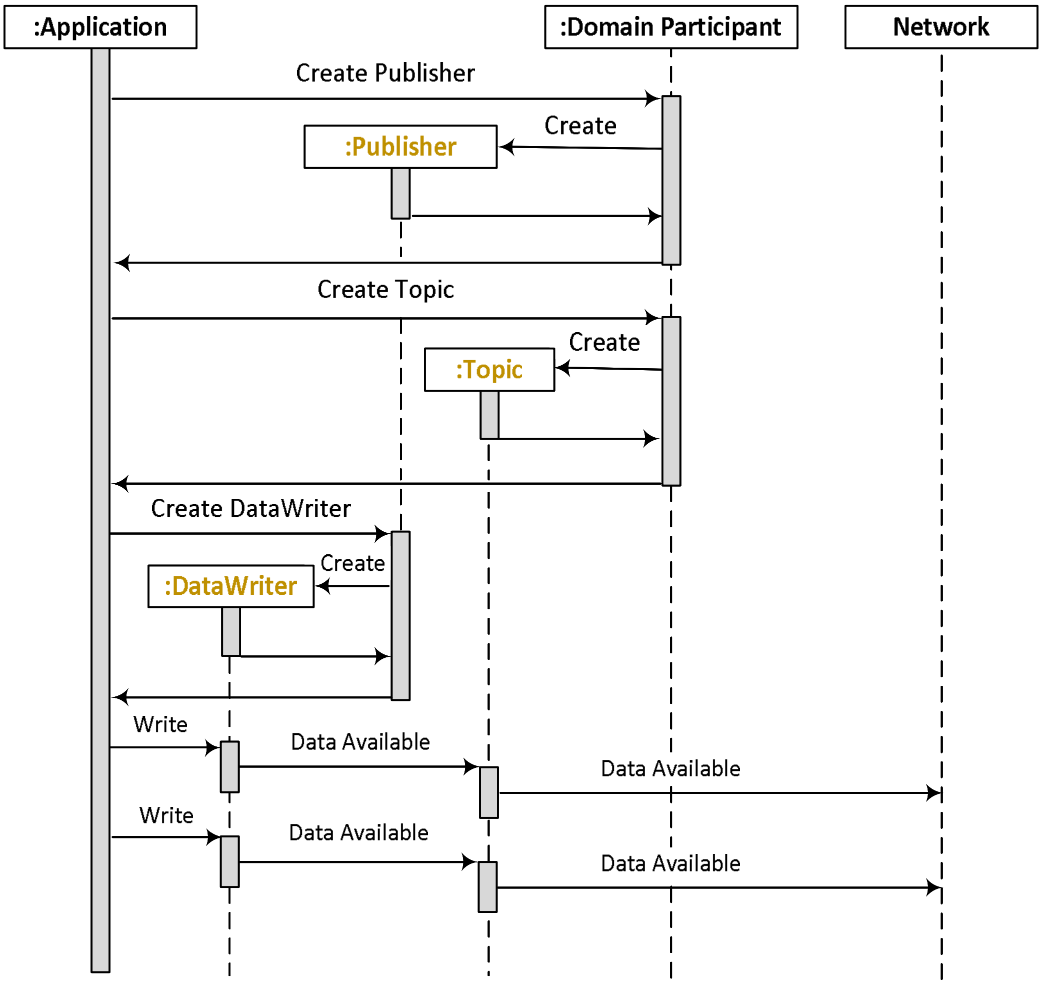

Figure 9. Publication Process over DDS Middleware

When a publisher's application produces an event, the event is written out by that publisher to a DataWriter. After that, the event is sent toward a subscriber application(s) by the Publisher component. The Subscriber component can monitor incoming messages while a subscriber's application is running at a node. Once the application subscribes to a topic of an event, the Subscriber should inform the local DataReader and SubscriberListener associated with that specific topic as depicted in Figure 8. Then the SubscriberListener will inform the application with the arrival of the event to reads the event through DataReader.

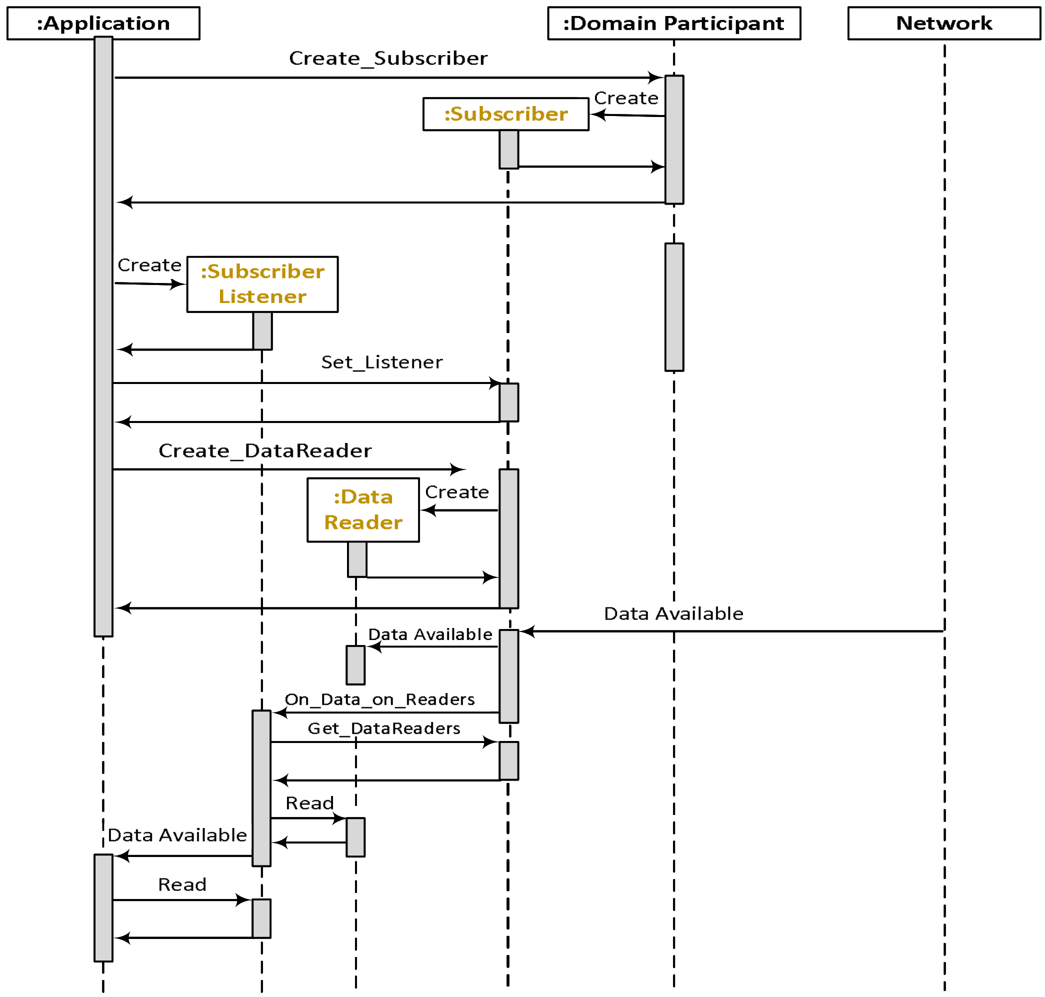

Figure 10. Subscription Process over DDS Middleware

A subscriber's application filters out all incoming events related to the subscribed topics by using a ContentFilteredTopic to derive Topic as shown in Figure 10. The ContentFilteredTopic component is used to determine a subscriber interest in the events that meet specific criteria. For instance, the subscriber's application can determine its interest in the events where topic is Temperature and the contents of that topic is in between 200 and 300 degrees. That can be performed if the criteria in the ContentFilteredTopic component is “Temperature > 200 AND Temperature < 300”.

1.2 WSANs Characterization Parameters

In this section, we discuss some WSANs characterization parameters that help developers to classify different WSAN applications according to their requirements. For example, Table 2 depicts the acceptable delay in some services. Generally, there are different parameters that characterize/identify WSAN applications. These parameters can be categorized in to six classifications: 1) communication and traffic, 2) service, 3) service components. 4) network. 5) node and 6) operation environment [14, 15]. In this study, we concentrate only on the first two classifications since they have direct impact on the three TinyDDS approaches.

Table 2. The acceptable Delay for Some Applications

Service	Acceptable Delay (ms)
Audio Broadcasting	<150
Internet Relay Chat	<200
Telnet	<250
Enhanced Web Browsing	<400

The communication and traffic classification consists of a set of parameters that define the WSAN resources in order to support the network applications. In this classification, any WSAN application is either a Non-Real-Time or Real-Time application [16] in terms of the delivery requirements. For instance, real-time applications such as video or audio conferencing require delay of less than 150 milli-seconds [17].

For the service classification, the WSAN applications exchange data that can be transmitted in Low, Medium, or High traffic rate. Figure 11 shows some WSAN characterization parameters that will help us to present our recommendations for using DefTDDS, BLTDDS and HyTDDS approaches.

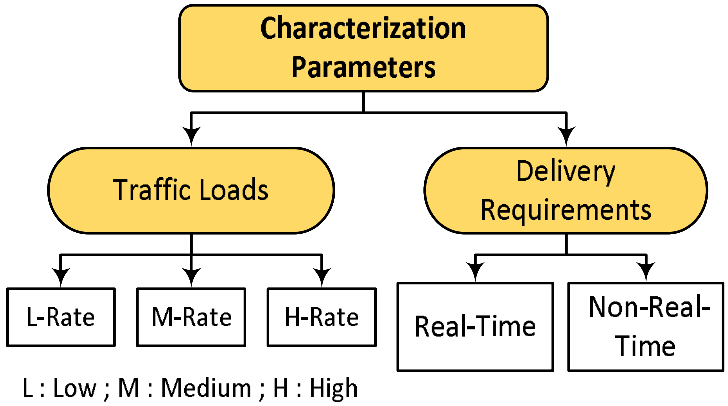

Figure 11. Some characterization parameters in WSAN

1.3 Problem Statement

Pub/sub interaction scheme is considered a suitable solution for large-scale and real-time distributed systems due to the data-centric architecture and decoupling properties. Features such as data centricity and decoupling properties can distinguish WSANs from other wireless networks. The pub/sub paradigm is able to use the limited resources in such environment efficiently [18]. It was proposed by many works to adapt pub/sub interaction system to WSANs [19, 20]. In WSAN, energy has been considered one of the most important resource constraint since sensors and actuators devices are battery-powered. Changing batteries becomes very difficult, mostly because of the harsh environment where these devices are deployed. Thus, energy balancing and saving is a critical subject in such systems. The energy consumption metric has not been considered in many pub/sub WSAN proposed solutions which has led to a lack of interest in energy saving techniques. Recently, the proposed standard-based solution in [21] has taken the energy issue into account and developed an energy consumption balancing protocol. However, it still needs some improvements in order to optimize the energy consumption balancing in such networks.

1.4 Proposed Solution

In this work, we enhance an energy aware protocol called EATDDS. The EATDDS protocol was developed using DDS standard which is a well-known pub/sub middleware. This middleware is used widely in enterprise networks. Moreover, DDS middleware provides a pub/sub abstraction that can simplify the tasks of communication. However, energy consumption factor has not been investigated thoroughly for DDS-based solutions in the context of WSAN. TinyDDS is a light-weight middleware that is a partial porting of the DDS.

To the best of our knowledge, EATDDS [21] is the only energy saving protocol developed on TinyDDS middleware. In EATDDS, a new RN will be selected in each cluster every 350 seconds regardless how much of energy can a node afforded. Although, this protocol can reduce the energy consumption in the network, it is still not optimal way to distribute and balance the energy over the network. in this work, we enhance the EATDDS protocol in order to make the energy consumption distribution and the network life time more efficient.

1.5 Research Objectives

This research mainly was conducted to use the network life time of the pub/sub interaction scheme in WSANs efficiently. This objective can be achieved by providing energy consumption balance over the nodes in the network. Specifically, the research objectives of this work are as follows:

a) Investigate the requirements and challenges of pub/sub model over WSANs.
b) A comprehensive study for the state of the art solutions of integrating pub/sub into WSAN.

c) A comparative study of TinyDDS approaches, namely DefTDDS, BLTDDS, and HyTDDS in terms of four metrics: (1) End-to-End delay (EED). (2) Packet delivery ratio (PDR), (3) Throughput and (4) Energy consumption.

d) Developing a new TinyDDS approach called Enhanced Energy Aware TinyDDS (E-EATDDS) that can be used to mitigate the single point of failure which occurs in centralized systems, especially in terms of energy consumption and to use energy in the network in an efficient way.

1.6 Methodology

The research methodology carried out in this thesis is divided into three main stages:

e) ***Stage 1:*** an extensive study of pub/sub systems in WSNs is conducted and we explore all technical challenges related to such systems. Also, we compare the requirements of efficient data dissemination. By the end of this stage, we expect to get a full understanding about the requirements and challenges of pub/sub model over WSANs.

f) ***Stage 2:*** we present a comprehensive study for the state of the art solutions of integrating pub/sub into WSAN. In addition, we provide performance evaluation of three TinyDDS approaches. First approach is DefTDDS and the other two approaches (BLTDDS and HyTDDS) are proposed by [21].

g) ***Stage 3:*** we propose and implement a new approach called Enhanced Energy Aware TinyDDS (E-EATDDS). The proposed solution adds enhancements to the EATDDS protocol.

1.7 Thesis Outline

The outline of this thesis is organized as follows. Chapter 2 introduces a comprehensive study of pub/sub communication systems and their properties, functions, strength points and shortcomings. In addition, this chapter presents TinyOS simulator (TOSSIM) as one of the prominent simulators for WSNs and its GUI. Finally, this chapter introduces the Online Energy Model which can be sued in TOSSIM. Chapter 3 provides a comparative study of three TinyDDS approaches namely, DefTDDS, BLTDDS, and HyTDDS. this chapter aims to evaluate the performance of the three approaches in terms of throughput, PDR, EED and energy consumption. Chapter 4 introduces enhancements applied to the EATDDS protocol and how to make it more efficient as an energy-aware protocol. In this chapter, we describe the final version of our proposed approaches called E-EATDDS. Moreover, we compare the proposed EATDDS with the previous TinyDDS approaches. Finally, Chapter 5 includes conclusions and future work.

CHAPTER 2

LITERATURE REVIEW

In the last years, many research projects and works have been devoted to explore and adapt Pub/Sub communication model to WSN. In this section, we present the Pub/Sub interaction model from WSN/WSAN standpoint and discuss different solutions suggested recently. Furthermore, we discuss the essential and important information about each approach including the main features such as architecture, components, and shortcomings.

2.1 Publish/Subscribe Systems in WSN

The first pub/sub communication model used over WSNs is Directed Diffusion paradigm [22]. This paradigm basically uses data centric protocol that can specify the ownership of data by attribute-based naming. Figure 12 shows the simple scheme for directed diffusion paradigm. The subscriptions are broadcasted through the network and called interests. Once the interest is broadcasted, the gradients are set up from the source to the sink and the matching process is conducted locally by each node. If the source has the requested data, it replies back with data to the sink. Otherwise, the requested data will be propagated throughout the network. Since directed diffusion scheme uses distributed matching process, it can distribute the consumption of energy equally and avoid the problem of centralized processing. In contrast, each node has to store and process the interest in the same way which add memory and processing overhead. The data structure is represented in pairs of attribute-value in order to get particular information. The direction state that points to the sending node is hold by a gradient associated with each requested interest.

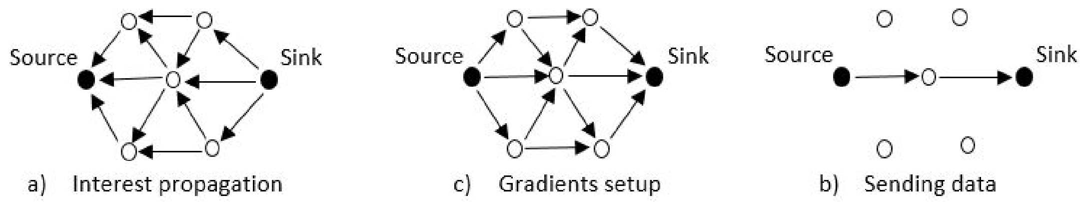

Figure 12. Simple Directed Diffusion Scheme

Another pub/sub model for WSN is Mires [23] that was proposed to ease the WSN applications development. This middleware is implemented on top of event-based TinyOS [24]. In Mires, nodes send or advertise their available topics to the Rendezvous Node (RN) in order to be sent to the user applications. Afterward, each application can select the topics of interest from the available topics on the RN and then broadcast the subscription messages into the network. Once the subscription messages are received by the sensors, they do the matching process to send back the requested data to the applications according to their interests. This middleware can reduce the transmission overhead by using an aggregation service. The architecture of Mires is depicted in Figure 13. As noticed, it provides the distribution manner over the nodes.

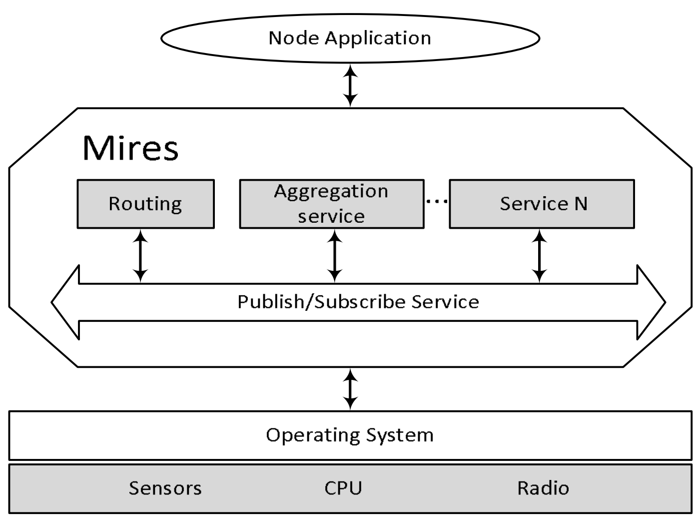

Figure 13. Mires Architecture

Nevertheless, Mires has some drawbacks such as 1) The behaviour of the network is controlled by a single node (RN). 2) The actuators are not supported.

The studies in [10, 25] discuss an IBM pub/sub protocol named Message Queuing Telemetry Transport (MQTT). It was proposed by Standford-Clark and Hunkeler, in 1999. MQTT is a lightweight and simple messaging protocol suggested to meet low-bandwidth constraint and unreliable networks. As a result, the work in [26] suggested an extended version of MQTT to overcome lightweight properties and mainly to be adapted to the infrastructure of WSNs. In MQTT protocol, the pub/sub service is called the notification service. This service is located in brokers using the original MQTT protocol and sensors software is kept simple; the MQTT-S architecture is shown in Figure 14.

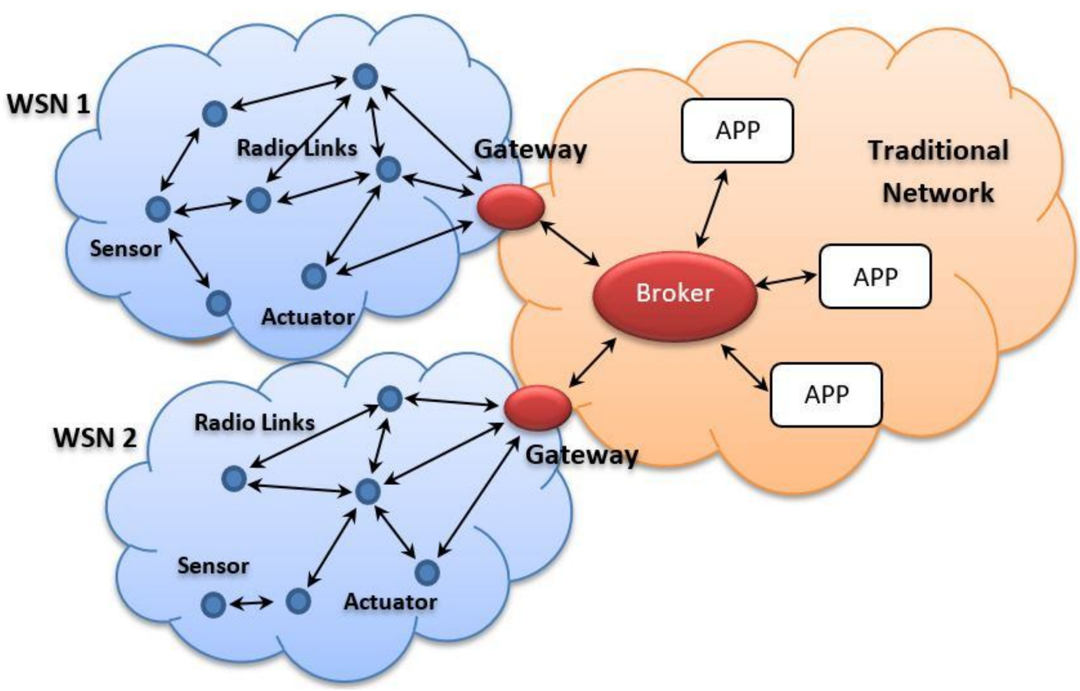

Figure 14. MQTT-S Architecture

Sensor and Actuator devices allow deployed devices in the network to transmit data to the closest gateway by using the collection tree protocol (CTP) [27]. QoS is supported in MQTT at three levels as follows:

- level-0: it provides the best effort service (no acknowledgment or retransmission),
- level-1: it allows the sender to keep retransmitting messages till it is acknowledged by the recipients.
- level-2: it allows retransmitting, but with no redundancy.

However, this approach has several drawbacks such as bottleneck and single point of failure which appears in the centralized approaches. Moreover, there is more delay occurs during the translation process between the gateways (MQTT-S) and the broker (MQTT). As a result, this protocol may not be suitable for real time applications.

Researchers in [28] proposed a content-based pub/sub service to WSN by using a component-based middleware. This middleware was able to simplify the composition and selection of each component. Consequently, the designer could adapt the service by making several orthogonal choices namely, 1) the components of communication protocol for notification delivery and subscription. 2) supported data attributes and 3) collection of service extension components. This scheme uses an attribute-based naming scheme augmented with metadata information in order to send control information to the publishers. The service extension components (SEC) can be reused in various platforms and applications because they are decoupled from the TinyCOPS. There are two different types of SEC that can be supported: 1) Communication SEC (CSEC) used to add services to the communication protocol, and 2) Attribute SEC (ASEC) used to add services to the endpoints. The framework high-level decomposition is depicted in Figure 15.

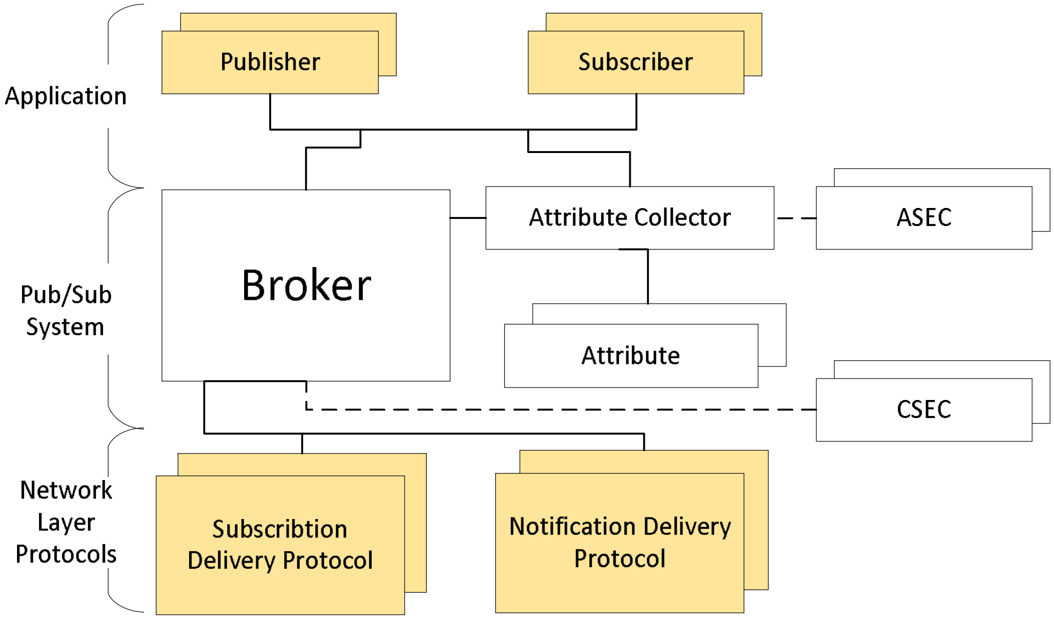

Figure 15. TinyCOPS architecture

The study in [29] presents a pub/sub middleware named PS-QUASAR. This middleware provides a high-level programming of pub/sub model and it can provide QoS such as reliability and priority. Furthermore, all nodes over the network are potential publishers. This model uses multicast technique (many-to-many exchange) to support exchanging messages over a fully decentralized manner. It can connect multiple publishers and subscribers transparently and ensure that the requirements of QoS are met. PS-QUASAR consists of three modules namely, maintenance protocol, routing module and the programming module as shown in PS-QUASAR architecture, Figure 16. The job of maintenance protocol is to discover pub/sub end nodes and to create links among neighbour nodes. The routing module uses information provided by the previous mode to route the events. The programming module is used to provide a topic-based pub/sub API allowing developers to evolve different WSAN applications on PS-QUASAR middleware. Though PS-QUASAR is robust, QoS-aware, and energy efficient protocol, it might have a memory space issue.

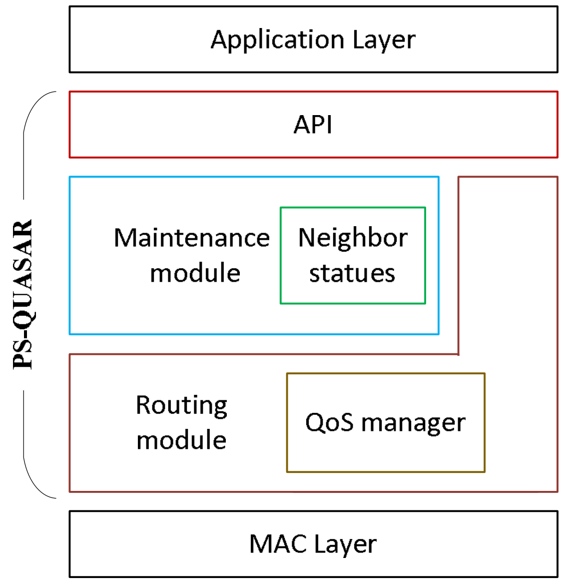

Figure 16. PS-QUASAR Architecture

PUB-2-SUB+ [30] is a content-guided pub/sub approach. It depends on content-guided routing to provide better efficiency in terms of storage and communication cost. A naming scheme [31] proposed for content-based pub/sub systems for WSN is used in this approach. In PUB-2-SUB+, a set of m spanning trees is stored where each tree is rotted at a different node in the network. Each root node is a dedicated reliable node and it is placed randomly in the network. Each tree has a naming tree where each node is represented as a binary-string name. The naming scheme is used to assign a "zone" for each node. The zone consists of the binary strings that starts with node’s name; not with any child node’s name. A query is applied to subscribe to a random tree while an event is used to publish to all the trees.

A query and an event in PUB-2-SUB+ take the form of an interval of binary strings and a binary string, respectively. If we choose a tree randomly, a query is routed to any node within the overlapping area (i.e. overlapping between a node's zone and the query’s interval). Each tree allows an event to be published to the node with the longest prefix name of the event string. Generally, even if there is some failure or disconnection in such approach, an event can easily find another way to the matching queries due to the multiple paths.

TinyMQ [32] is an improved version of the PUB-2-SUB+ solution. It is a content-based pub/sub middleware which adds content-based routing and uses multiple sinks to avoid the congestion. A node in TinyMQ can work both as a producer/publisher and as a consumer/subscriber of information. Nodes exchange two types of messages, events and queries. This approach comprises two layers: overlay layer and pub/sub layer. The overlay layer assigns a unique virtual address to each node so as to maintain a naming structure. The pub/sub layer can provide mapping and the functionality of routing for different events (subscription, publication and notification). The message mapping and routing technique depends on queries and events to determine the paths of the corresponding subscription and notification. The routing mechanism relies on the nodes' virtual addresses. For events and queries, the message mapping allows the initiated nodes to select the rendezvous nodes (RNs) and then, send corresponding messages to these nodes (RNs) through message routing mechanism. Afterward, the RNs decide whether to send the events to the subscribers or no by checking the events against the queries. TinyMQ supports content-based routing and none location-based information dissemination. It provides interoperability between the nodes in WSN. Nevertheless, it does not support QoS and actuators to WSN.

Quad-Pub/Sub approach [33] is one of the proposed solutions as a pub/sub system for WSN. It supports the resource-aware routing operation transparently by using the location-based addressing mechanism. The main goal of this study is to increase the efficiency of the network life time by distributing the load of routing over multiple paths and reduce the cost of communication through using the shared event dissemination paths. A localized resolving algorithm is used in Quad-PubSub scheme to provide a simple operation and distance calculations. Such algorithm can resolve the sub/unsub operations iteratively. Also, it sets up paths where end-point publishers/subscribers are not involved. The publishers and subscribers are decoupled by a set of intermediate nodes, called Event Brokers (EB). The network is divided into sub-areas and each EB is responsible for one area. The EB receives the subscriptions and does the matching process with the published events in its area. Afterward, it serves the interested subscribers according to the matched events. Although EBs distribute the data dissemination to give communication load balance, they die before the other nodes affecting the network connectivity. furthermore, this approach does not support QoS and there is no energy consumption evaluation in the study.

TinyDDS [34] is a pub/sub solution for WSN that was adopted by OMG DDS standard. It is a lightweight pub/sub middleware implemented over TinyOS code. TinyDDS allows an application to provide interoperability across the WSNs boundary to access networks without considering their protocols and programming languages. Furthermore, it can allow applications in WSNs to have fine-grained control at application level and middleware level. According to the dynamic conditions of the network, TinyDDS performs event publication adaptively and can balance its performance between conflicting objectives autonomously. TinyDDS has added two main contributions to WSNs namely: 1) interoperability with access networks, and 2) flexibility to customize and non-functional properties (e.g. routing and data aggregation).

TinyDDS provides two implementations: (1) one for TinyOS platform (e.g. Mica Z, Mica 2, or TelosB sensor nodes) and (2) one for Sun Microsystem platform (e.g. SunSPOT sensor node). The architecture of TinyDDS and its main components are shown in Figure 17. TinyDDS middleware plays a major role as a pub/sub system for WSN providing services and support, but because DDS proposes only a pub/sub model, it is still the responsibility of developers to provide QoS properly [35].

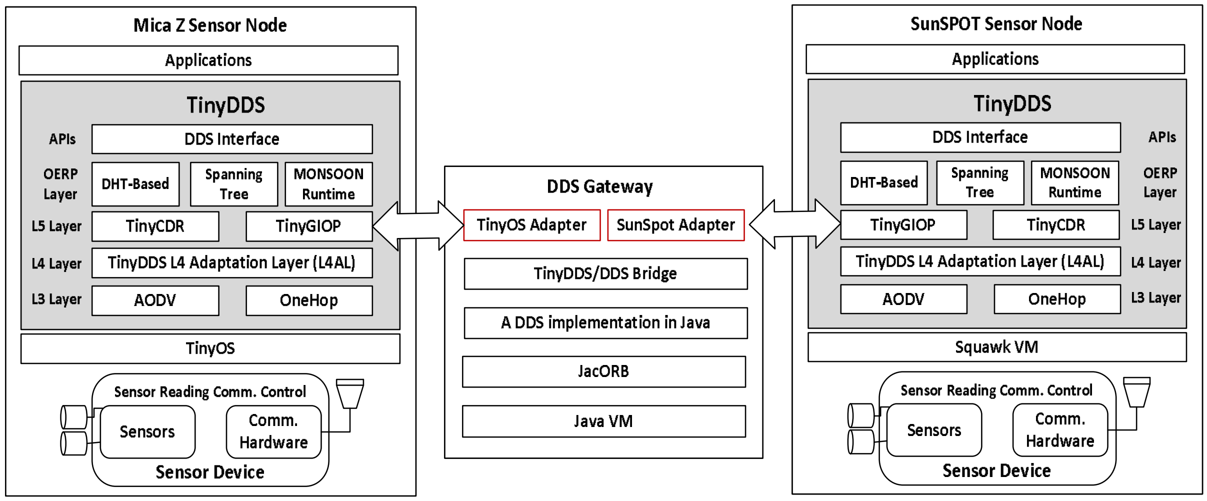

Figure 17. TinyDDS Architecture on MicaZ and SunSPOT Platforms

EATDDS [21] is an energy aware pub/sub protocol for WSN. It is the latest enhancement for TinyDDS middleware as a pub/sub energy aware scheme based on DDS standard. This protocol can reduce the distances between the publishers and their interested subscribers by using nodes' location. The main goal of EATDDS is to use the energy efficiently in the network in order to increase the network life time. The EATDDS protocol divides the network into clusters where each cluster consists of one RN. Basically, each RN represents only one topic with all the relevant publishers and subscribers forming a cluster. Although, the EATDDS can provide a mechanism to distribute the energy consumption in network, it is still imbalanced energy consumption. It selects a new RN for each cluster periodically while the old RN might still have the maximum energy in that cluster. This approach is not an optimal solution to the energy consumption problem since selecting the RN does not depend on the energy information.

The study in [36] proposes a topic-based pub/sub middleware for WSN. It is a decentralized pub/sub model which uses the notification service in a distributed fashion. Meaning that, each node should manage its subscriptions by using its own routing table and publish the generated data depending on these routing tables. This approach has two main phases, subscription messages distribution phase, and publication messages distribution phase. In this system, the framework along with the forwarding mechanism on the application layer are responsible for the messages distribution over the network. The implemented application uses the routing tables to perform the forwarding mechanism. This middleware is able to decrease the transmitted messages which leads to reduce the energy consumption. However, using routing tables on nodes leads to a memory overhead problem.

2.2 Publish/Subscribe Model characteristics

2.2.1 Data-Centric Architecture

As a vital feature of WSANs, data-centric differentiates other wireless data networks from them. It offers productive utilization of their finite resources and corresponds well their nature [37, 38]. In WSANs, the application is interested in the data gathered from the examined physical environment and not in the identity of the sensor. Of late, many applications might be interested in various types of data using the same WSAN infrastructure, while the traditional single sink WSN applications were created mainly to support one application for each network. As an illustration, for a building surveillance application system, there might be the demand to simultaneously survey the cracks in the wall, intensity of light, and building temperature. Additionally, actuators might be contained in it to support the stress, for instance, adjusting the temperature of the building by unfastening the valve available in the cooling setup. Usually, the concept of data-centric producer/consumer (pub/sub) interaction model is the result of this type of applications. The subscribers are mainly concerned with the incoming information transferred by the publishers, and they do not know the origin of the data in terms of the network address.

2.2.2 Many-to-Many Interaction Scheme

The sensor network based applications are migrated by WSAN and multiple sinks sensor network applications from one-to-many to many-to-many communication model. Data is expected to flow in both directions in these new applications from sensors to sinks or actuators and the other way around. For instance, publishing the data collected for the monitored area is the primary role of the sensor while that of the actuator is to act as the subscriber subscribing to the sensor data to be examined and to perform some necessary response. On the other hand, in order to gain access to the control data coming from the sink (e.g. wakeup, sleep, or configuration data), the sensor also acts as a subscriber while the actuator acts as a publisher in order to transfer the information to the sink nodes. Therefore, the extensive disseminated sensors and actuators alongside many-to-many interaction demands are accomplished by pub/sub interaction model, where it is fundamentally a many-to-many interaction paradigm [9].

2.2.3 Heterogeneous network

New variabilities of sensor platforms are present in the field of industry nowadays, as result of the absence of standards in WSN technology [39]. Hence, in order to satisfy the demands of the applications, a securely coupled application is created where the detailed information regarding the layers of focused platform of the network should be aware by the creators. In addition, integrating different platforms or integrating the WSN to the Pub/Sub-based enterprise networks is a difficult and complex task [10]. Finally, after vigorous efforts by the developers, a tightly coupled complex applications are created, in which these applications tend to be complex, disposable, immovable and even uneasy to upgrade. These problems are alleviated by the introduction of the pub/sub middleware by applying a middle layer within applications and essential platforms. That will simplify the development of the applications and grant them more portability, interoperability and upgradability as shown in Figure 18.

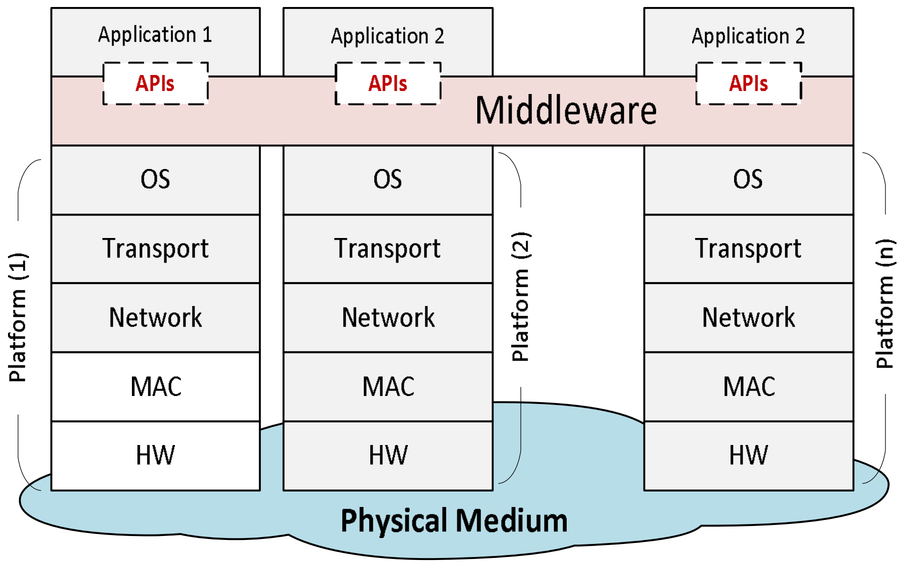

Figure 18. The Middleware Layer Between Applications and Underlying Platforms

2.2.4 Dynamic Network Topology

In spite of mostly being stationary, sensor networks appear dynamic in many situations as listed below:

- When the nodes are connecting to, or departing the network as a result of software or hardware failures present in the network links or node.
- Because of software errors, new applications might crash or attached at the consumer monitors.
- The node state constantly changes to sleep or deep sleep modes from active mode for saving energy, where it might connect once more through a different network address.

- The addresses of some of the protocols of the WSN network are changed from time to time.

As a result of this dynamic behavior of the network, the best solution choice for such network type is the Pub/Sub interaction model. Since in this interaction paradigm, the data is saved in queue structure called buffers and delivered whenever a connection is established. Furthermore, the essential network details are hidden from the application by the middleware of the Pub/Sub model to alleviate the constant changes in network addresses when nodes exit and connect to the network.

2.3 Components of Publish/Subscribe Middleware

The main elements of pub/sub application in the framework of sensor-based networks are described in this section. There are five major components in a pub/sub setup as presented in Figure 19, namely:

1. the programming abstractions and application programming interfaces (APIs).
2. end nodes (publishers and subscribers).
3. event/query (pub/sub) messages.
4. pub/sub service (notification service).
5. QoS mechanisms which can be compatible with pub/sub systems.

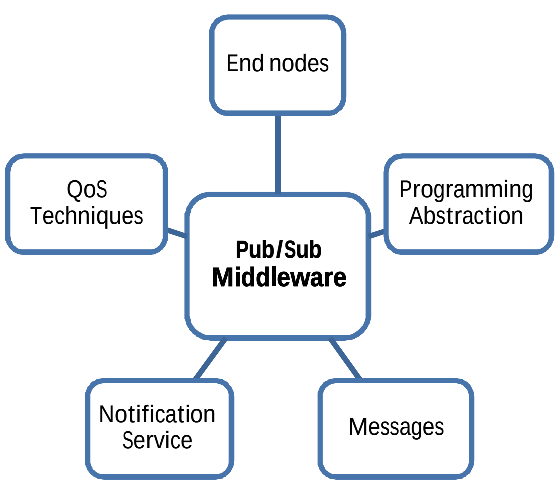

Figure 19. Publish/Subscribe Middleware Components

2.3.1 End Nodes

There are usually two end users for any communication system, the sender and the receiver. They are called the publisher and the subscriber in a pub/sub system, respectively. Events are created by the publisher and are sent to the interested subscriber through the notification service. In situations where there is no interested subscriber, the notification service stores the event by means of event table until either its expiry date is reached or a new subscription is received. Subscriptions are created by the subscriber and are sent to the notification service. A matching process is then activated in the notification service to look for a matching event. In the case where there is no matching event, the subscription is saved by means of subscriptions table till there is a matching event or it expires. As shown in Figure 20, the system (publishers and subscribers) in WSAN consists of four major entities: the application or the end-user, sensor, sink, and actuator.

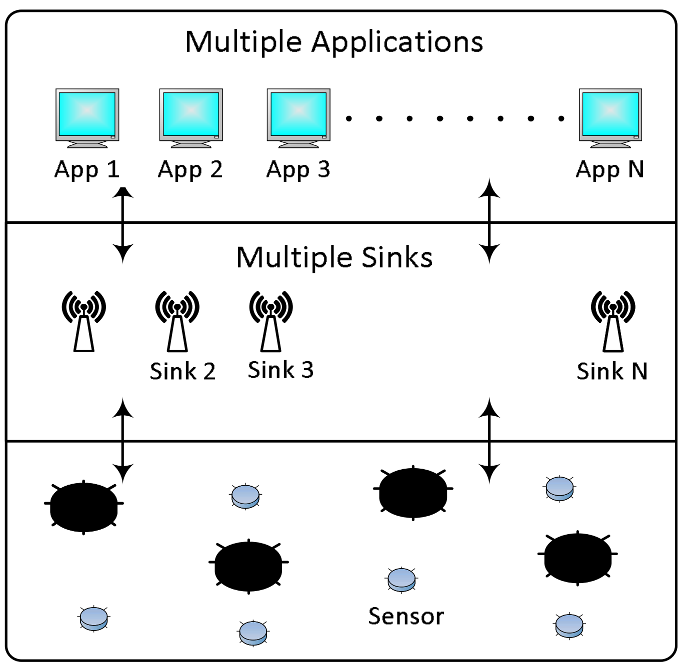

Figure 20. The Architecture of 3-Layers WSAN

These four entities are spread over three virtual layers, with each layer having distinct SW and HW capabilities. Consequently, different varieties of pub/sub system middleware are spread over the three layers. Considering the key function of the SA device, the sensor and the actuator would be considered as the publisher and the subscriber, respectively. However, in real-life scenarios, all four entities can act as publishers and subscribers simultaneously.

2.3.2 Programming abstractions

Programing abstractions are presented in the form of APIs so as to minimize the complication and improve the efficiency of the WSAN systems expansion. For instance, the main middleware APIs in event-driven pub/sub applications [40] are delivered to develop, publish, subscribe and unsubscribe a particular event. The programming abstractions here are developed to facilitate the improvement of the application by obscuring the heterogeneity together with the complex and detailed information about the fundamental layers of the network from the developers of the application [41]. Two major levels of abstractions exist in the framework of WSAN system applications, namely: the node and the system levels. The node level provides a fine-grained control to the developer on the network system [42], where the action and support of the individual SA devices can be programmed. Thus, this level of abstraction supports the developer to build more efficient WSAN applications in terms of resource allocation and power consumption. The WSAN abstracts to a single virtual system, at the system level, with global behaviour that renders the task less difficult but with less control to the SA devices [43]. In this regard, the developers build a centralized program, where they focus more on the functionality of the whole system neglecting the coordination mechanism of SA devices.

2.3.3 Messages

Three main types of messages are present in the pub/sub communication model, namely: (1) event/publication, (2) query/subscription and (3) advertise. The advertise type of messages are used in the advertisement of events prior the publication [10, 23]. Such messages, developed by the system application, comprise of the payload and header message, and usually consist of key fields located within the header of the message such as issuer, an identifier, and certain fields pertaining to the QoS parameters including expiration time, priority and deadline. There is variation of the format of the message from one implementation to another, for instance, the message is represented in the form of an array of bytes [44] or utilize a set of varieties such as XML or text [45], or permit the developer to develop a custom structure type of message [46]. The general format of the message in the pub/sub systems [47] founded in WSN, consists of the packet header (2-4 bytes) and the data payload (n bytes).

There are varieties of ways that subscribers express their interest in events from the standpoint of event expressiveness. They range from one implementation to another and directly influence the design and algorithm utilized for the implementation of the notification service. Figure 21 shows the four regular schemes used to express events in the pub/sub paradigm. These schemes are topic-based, channel-based, type-based and content based. These will be referred in this study as pub/sub application systems in that they will be used to differentiate between the monitored WSAN pub/sub application protocols.

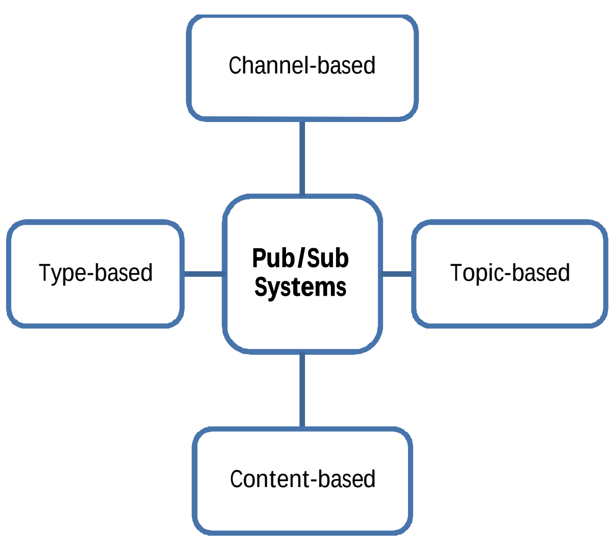

Figure 21. Pub/Sub schemes

2.3.3.1 Channel-based system

This system functions to group the events or notification under various channels, in a way that only the channel which contains the interested events will be subscribed to by the subscribers. In this approach, the events published are without topic names. However, with each published event, a channel-id is inserted. Hence, to publish an event to a particular channel, denotes that this event is broadcasted to all subscribed entities to that channel and the other way around. A queue structure is utilized in Java Message Service (JMS) [48] to apply channels within the broker of the notification channel. For instance, if an event is published by a publisher with a specific channel-id or queue-id, the queue-id linked to this event is searched by the broker and inserted to a queue using first-in-first-out (FIFO) approach. The subscriber subscribes to the channel on the other side by specifying or providing the queue-id which can also be called the queue name. The events received from the publisher is immediately routed to the subscriber with the same queue-id by the broker. Upon receiving the events by the broker, a check will be made to know whether there is no queue having the same event queue-id. A fresh one will then be created if there is no same event queue-id. As far as we know, there is no channel-based pub/sub solutions for WSAN/WSN suggested in the open literature. Meaning that WSAN applications are limited in resources and inclined further to content-based systems that considerably reduce whole network traffic and as a result, reduces the bandwidth and the consumption of energy. Other instances of industrial utilization include the CORBA event service and CORBA notification service [49, 50].

2.3.3.2 Topic-based system

The concept of the channel-based approach or scheme is extended by this system through the addition of more characterization and classification for the content of an event [51]. The topic-name matches the channel-id, which results in the formation of a logical channel which links all the subscribers interested in a certain topic to the publisher. In the development stage (static subscription model), a fixed number of topics are produced, and a distinct topic-id is assigned to the notification by the publisher utilized by the notification service during the process of matching to acquire the matching subscriber that is interested in the published topic. A hierarchical approach is utilized to advance further in the classification of the content of an event to make this scheme more expressive [52]. The topic in this way could be divided further into sub-topics through the utilization of a tree structure. A unique key is assigned to each individual topic and in each topic, are numerous cases where, each case inherits the attributes of the topic and recognized by a topic main attribute that could be a field inside the topic. Several research suggestions regarding the topic-based pub/sub middleware application in WSN are available in open literature; examples include PS-QUASAR [29], TinyDDS [34] and Mires [23]. The ability to easily utilize the current group-based multicast procedures, such as IP Multicasting [53] through the specification of each topic to a multicast group is the advantage of this system.

2.3.3.3 Type-based system

This is another event subscription scheme [54]. In type-based scheme, events with a particular structure or type are subscribed by the subscribers rather than subscribing to topic name. For instance, If we have a structure called product which has some attributes such as name and price. Then the subscriber subscribes to the structure name (the product) rather than subscribing to the name or the price of the product in our example. Hence, all the events with the same structure will be received by the subscriber.

2.3.3.4 Content-based system

This is a fine-grained control system that enhances the extent of expressiveness of the various subscriptions. Here, the subscriber expresses concern in a more accurate and dynamic fashion, unlike topic-based, which offers constrained and static expressiveness. Furthermore, the content of the event published in topic-based scheme is not visible to pub/sub application service apart from the topic-id, while the content of the event published in content-based is made visible to the pub/sub application service. Consequently, the topics that is not interested by the subscriber can be filtered out by putting restrictions on the content of the subscribed topic. It is permitted to subscribe to an event in the content-based systems with applying particular constraints through the use of operators for comparison (e.g. <=, =, =>, <, >). A topic (e.g. a specific temperature of a machine) is published by the sensors and various patterns from the topic published is received by each subscriber (actuators & sink) based on their predetermined interest as shown in Figure 22.

Tradeoff exist among the behavior of high performance regarding resource consumption and delay, and the extent of expressiveness and it is not easy to develop and implement such filtering algorithms. Many pub/sub architecture algorithms are suggested by many researchers to reduce the overhead and expended time by the process of content filtering [55, 56]. In systems with limited resource like WSANs, the process of event filtering considerably influences their performance, particularly for real time applications. In one way, the overhead processing is increased by such process and more end-to-end delay are added. In another way, it minimizes the overall bandwidth utilization which results in increased performance in the network regarding delay and throughput. A number of content-based protocols has been suggested for WSNs, for instance, the TinyDDS [34], MQTT-S [10] and TinyMQ [32]. It is always advantageous to utilize topic-based scheme for applications where the space of the event can be distributed to inadequate set of possible discrete values, in order to prevent further overhead as a result of the content filtering system.

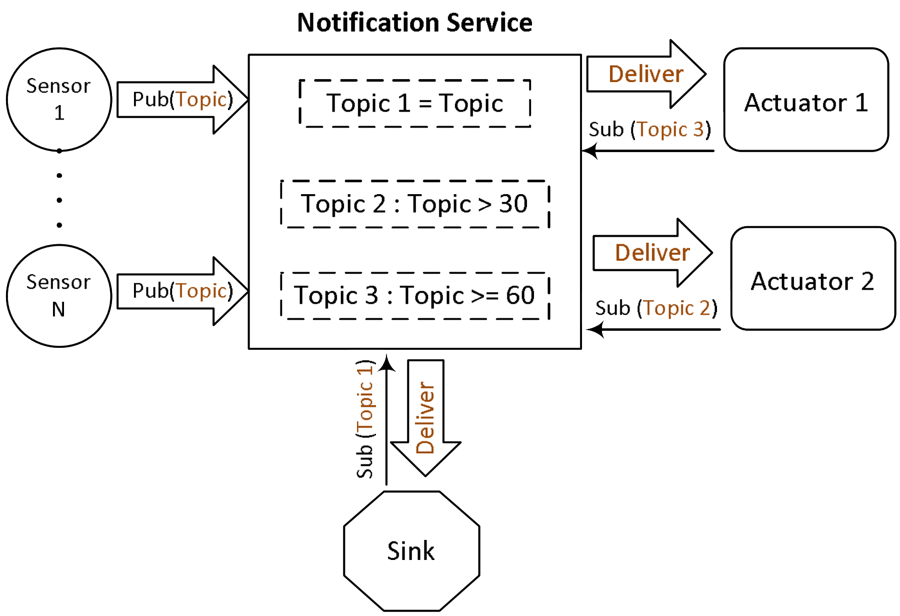

Figure 22. Content-Based Interaction System

2.3.4 QoS Mechanisms

Quality of service (QoS) support is guaranteed by the communication medium. These guarantees differ strongly between various systems [9]. For any WSN middleware, one of the advanced features is the support for QoS. Providing QoS support in pub/sub is a difficult task [57]. This is because unlike the straight link that exist between the receiver and the sender, the decoupling features found in the pub/sub paradigm render the system less deterministic. The most popular QoSs offered by the pub/sub paradigm in WSAN are discussed in chapter 1.

2.3.5 Notification service

The task of data distribution in pub/sub systems, lies on the component called the notification service (NS). This is considered as the main part of the pub/sub middleware coordinating among the subscribers and publishers. Interaction of the NS with the subscribers and publishers is achieved through specific operations as shown in Figure 23. Publish () and advertise () functions are used by the publishers to publish and advertise new topics. Also, the subscribe () and unsubscribe () functions are used by the subscribers to subscribe and to unsubscribe to a specific topic.

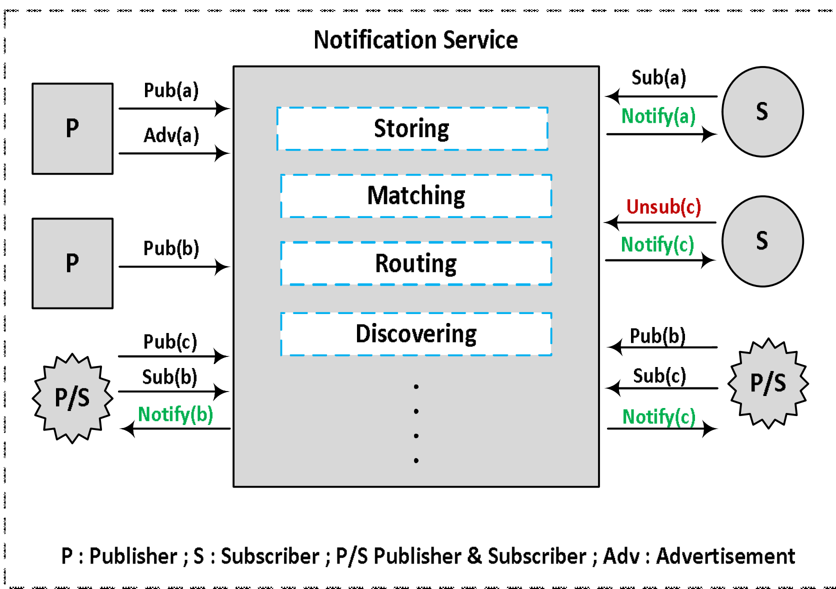

Figure 23. Notification Service Architecture

Finally, the notify () function is used by NS to notify the subscriber having a matched topic. Storing the publications and subscriptions, managing pub/sub QoS, discovering the publishers and subscribers, event filtering based on the subscription constraints, are some of the main services provided by the NS. In the resource-constraint systems including WSAN, these services are still a subject for further study. There are two main services of the NS [58], (1) the matching service, where it matches the publications with the desired subscription and maintaining that through the use of matching tables; (2) the routing services, where the matching publications are routed from to the relevant subscriber from the publishers.

2.4 TinyOS

TinyOS is an open source operating system which has been designed for low-power wireless devices [24] such as Sensor/Actuator (SA) devices, personal area networks, ubiquitous computing, and smart buildings. It makes a feature of a component-based architecture that enables rapid implementation and innovation while minimizing code size. The library of TinyOS's component involves distributed services, sensor drivers, network protocols, and data acquisition tools. TinyOS is used widely in simulation for academic and industrial purposes. It can be used to test and develop various protocols and algorithms.

2.4.1 TinyOS Simulator (TOSSIM)

TOSSIM is a discrete event-based simulator which can bridge the gap among algorithms and implementation providing scalability and high fidelity. TOSSIM is one of the most accurate and well-known simulators for WSNs. It is able to capture the mote behaviour at a very fine-grained control over application level and middleware level. TOSSIM can simulate a large number of motes at once. It provides discrete event simulation that allows a very event driven execution provided by TinyOS to go well [59]. In general, sensor networks follow the nature of an event-driven allowing motes to be tested individually which is insufficient. A program should be tested in rich and complex conditions in order to catch wide range of interactions.

In WSNs development, deploying a large number of motes becomes a difficult task since most of the work will be spent on maintenance rather than development. Therefore, when a particular mote allocated in a remote location has been failed then dealing with this failure causes an overhead over the entire network. For testing purposes that overhead is not acceptable. This simulator is able to deal with such problems by providing reproducible and controlled environments, by supporting access to tools like debuggers, and delaying the deployment till an algorithm are understood and codes are tested. In TOSSIM, each mote unit such as Radio and MCU corresponds to one or more components.

2.4.2 Java TinyOS Simulator (JTOSSIM)

JTOSSIM is a Graphical User Interface (GUI) to TinyOS simulator. It allows a user to interact with TOSSIM by defining different simulation parameters such as radio settings, network topology, number of nodes and the debugging channels. Figure 24 shows an example of a network built using the JTOSSIM user interface. JTOSSIM software tool requires a fully configured TinyOS development environment and Java 1.6 or greater.

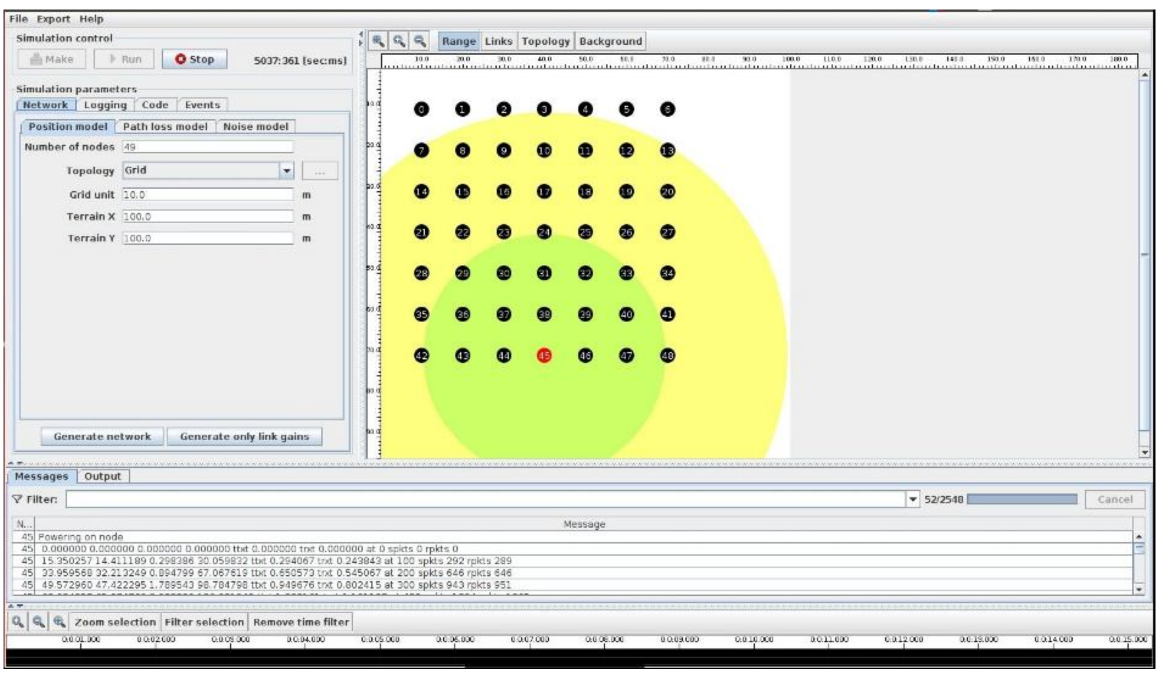

Figure 24. TOSSIM Geographical User Interface

2.5 Online Energy Model

TOSSIM does not support energy measurements that is a significant metric in the performance evaluation of WSN protocols. As a result, POWERTOSSIM [60] and POWERTOSSIMZ [61] are two energy measurements tools which have been integrated into TOSSIM to deal with this issue. POWERTOSSIM and POWERTOSSIMZ simulators are used for mica2 platform and micaZ platform, respectively. They track the power states of TOSSIM simulator's components such as Radio and Microcontroller Unit (MCU). Since POWERTOSSIM and POWERTOSSIMZ only support the mica series platforms and compute the final energy measurements after the simulation run, the Online Energy Model (OEM) [62] was proposed. OEM concentrates on the Radio and MCU components, allows to track the energy level of the network nodes during the simulation. Moreover, OEM was proposed mainly to develop the energy-aware protocol (EATDDS) [62]. The architecture of OEM is depicted in Figure 25.

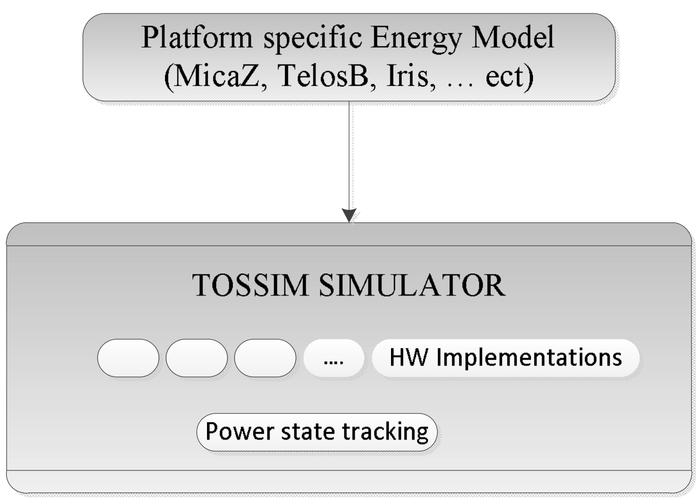

Figure 25. Architecture of Online Energy Model

2.5.1 Radio Component

The amount of energy consumed by this component is the largest compared to the other components in the mode. The micaZ platform uses CC2420 Radio Chip. In TOSSIM, here are three main interfaces provided by the Radio corresponding component, Send, Receive and Splitcontrol. The states of radio energy can be tracked using Send and Receive interfaces that can be found in the TossimPacketModel.nc component. The Radio component has Three main states that can be tracked, Send, Receive and sleep. Thus, the total energy consumption is the multiplication of the state duration (Δt) of receiving/ sending/ sleeping, the used voltage (V), and the consumed current of the energy state (I_{Pstate}) as shown in Equation 2.1. The Online Energy Model of radio flowchart is depicted in Figure 26.

$$EC_{Radio} = \Delta t * V * I_{Pstate} \tag{2.1}$$

2.5.2 Microcontroller Unit (MCU) Component

We need to track the time spent in each state of MCU power in order to calculate the energy consumed by the MCU. For each state, Equation 2.1 can be used also to compute the energy consumption of MCU. The Active and Idle states are two main states used to measure the MCU energy consumption. Moreover, the tracking code of the MCU power state was integrated with SimSchedularBasic.nc component. As long as there is tasks in the scheduler, The MCU in the Active state otherwise it is in the idle state [21]. The Online Energy Model of MCU algorithm is shown in Figure 27.

RT: Round Time
S: Send
R: Receive
ES: Event Stamp
ST: Simulation Time
D: Duration
AT: Active Time
ECs: Sending Energy Consumption
ECr: Receiving Energy Consumption
ECi: Idle or Sleep Energy Consumption
V: source voltage
Itx: transmission current
Irx: receiving current
Iidle: Idle current.
IT: Idle Time.

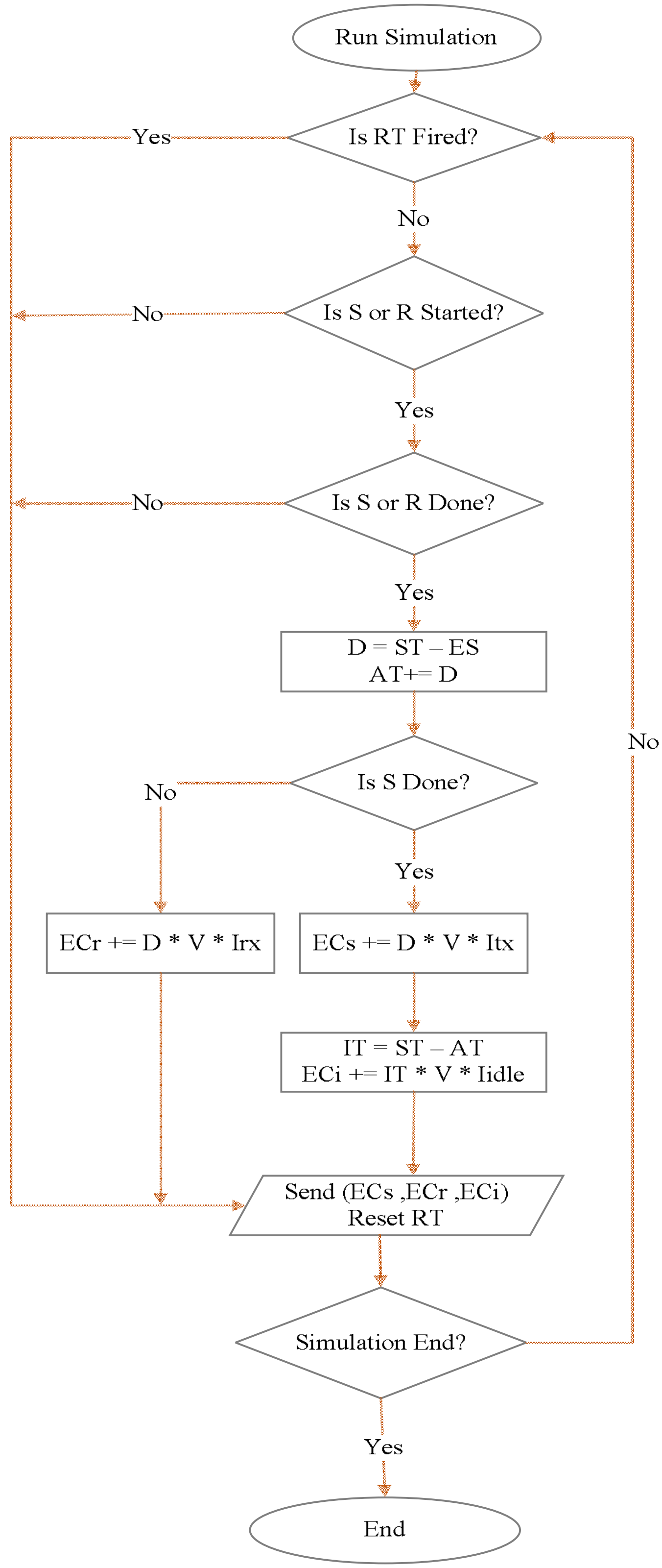

Figure 26. The OEM Radio Algorithm Flowchart

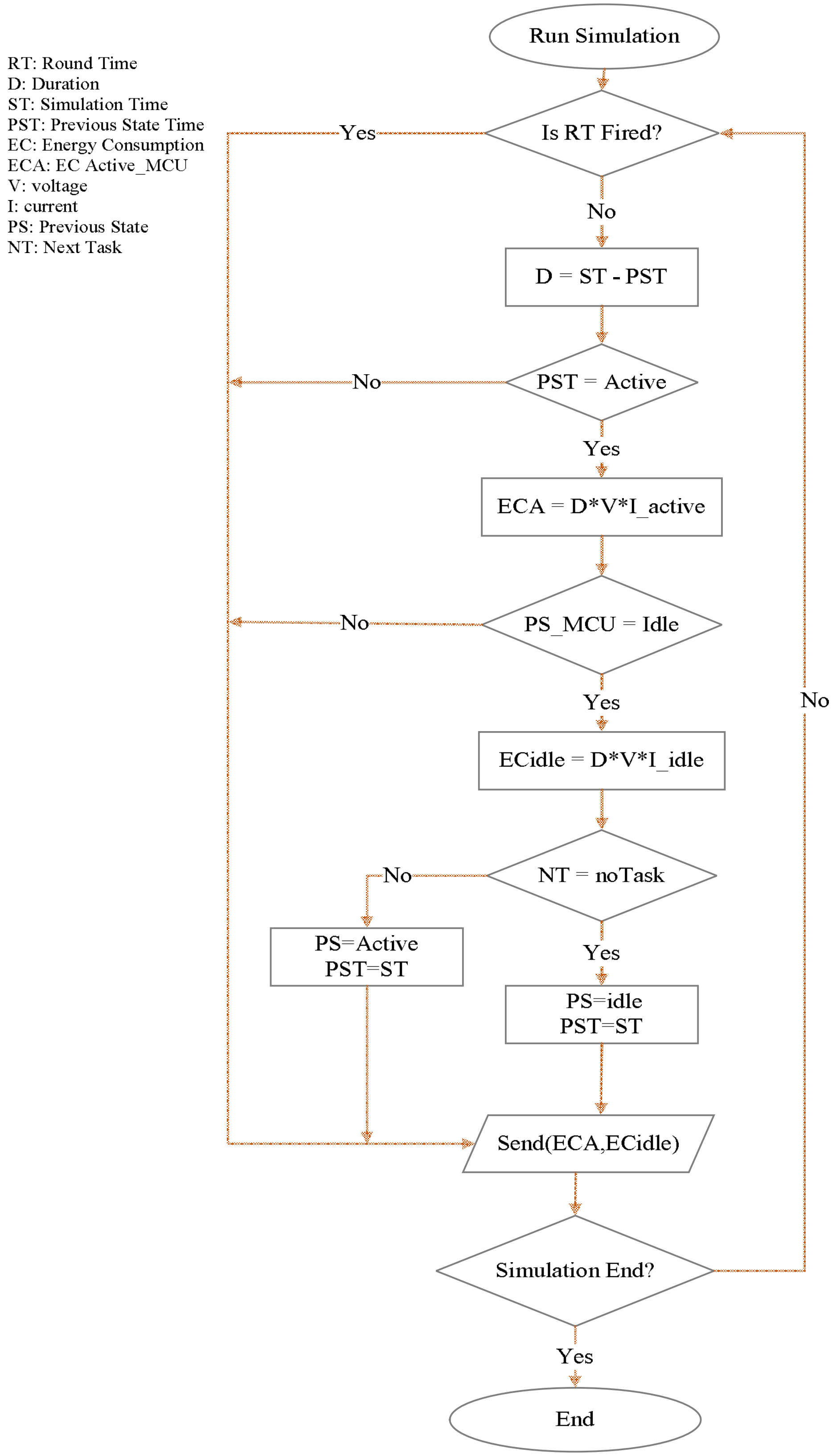

Figure 27. The OEM of MCU algorithm Flowchart

CHAPTER 3

Performance Evaluation of DefTDDS, BLTDDS and HyTDDS

This chapter presents three alternative solutions which can be used in TinyDDS to adapt pub/sub communication model to WSANs. They are Default TinyDDS (DefTDDS), Broker-Less TinyDDS (BLTDDS) and Hybrid TinyDDS (HyTDDS). DefTDDS uses the Rendezvous/Broker node essentially in order to exchange messages between publishers and subscribers. In contrast, BLTDDS eliminates the Rendezvous node (RN/Broker) totally during exchanging pub/sub messages while HyTDDS eliminates the RN only in the data dissemination phase.

3.1 TinyDDS Architecture

The architecture of TinyDDS [34] shown in Figure 17 on page 26 runs on each sensor node. TinyDDS middleware is operating in the transport layer and on top of the network layer (L3). It adds an improvement to the pattern of layer design proposed in [63] by dividing different functionalities into separated layers. Applications can use a set of DDS interfaces provided by TinyDDS at the top layer. These applications are able to disseminate events with associated topic to the network. When a subscriber in the network has interest on a particular topic, he/she can capture the events related to that topic. For routing events, DDS interfaces implementation runs on an overlay network. The layer of Overlay Event Routing Protocols (OERP) is used to provide various routing protocols that can perform the overlay network. OERP layer helps application developers to select suitable routing protocol which can meet their requirements. For instance, In WSAN which has limited memory space, the spanning tree routing protocol might be utilized to minimize the space of memory required to save routing table while the Distributed Hash Table based (DHT- based) routing protocol can be used to reduce the energy consumption of sensor nodes.

Routing protocols in OERP layer use the implementation of low-level network layer through TinyGIOP layer (L5). TinyGIOP is a session protocol (L5) implemented by TinyDDS to define message format used for exchanging among TinyDDS/DDS applications depending on General Inter-ORB Protocol (GIOP). It can support three types of message format namely, Request, Reply or CancelRequest. TinyGIOP allows exchanging data with DDS applications by interacting with the DDS Gateway. Only base stations connected physically to the DDS gateway can exchange data with that gateway. Moreover, in L5, another component called TinyCDR can provide interchangeable format of data to allow exchanging data among various implementation of DDS or TinyDDS. To send/receive data to/from other nodes in WSANs, TinyGIOP uses TinyDDS L4 Adaption Layer (L4AL). L4AL is a transport layer interface that allows TinyOS to operate with any network layer (L3) and MAC layer (L2) protocol. TinyDDS middleware has been implemented to operate different pub/sub communication protocols. To the best of our knowledge and beside the default TinyDDS, there are two protocols (BLTDDS and HyTDDS) [21] were integrated recently with TinyDDS middleware to adapt the pub/sub communication model to WSANs. Generally, the participants in these three protocols have two main phases: Discovery phase and Data Dissemination phase.

DHT-based event routing protocol provides an efficient lookup service. It organizes data into pairs of (key, value). A key is the product of a hash function which balances the distribution of keys in the entire network. In DHT, each node has a unique identifier (ID) that must belong to the same space of the used hash function output. To recover data of a particular key, it is helpful to identify the node storing that key. Thus, first, the key should be hashed by the searching node. Then the query is routed by the searching node to the node that has the closest ID to the key hashing.

3.2 TinyDDS Approaches

Generally, the participants in such pub/sub systems has two main phases, Discovery phase and Data Dissemination phase. When a node joins a network, it sends subscription messages to start the discovery phase. The node keeps sending subscription messages to the middleware till it is recognized. Afterward, the node switches to the data dissemination phase and the middleware can start sending to the joining node (i.e. the subscriber) according to the data of interest. In the literature, there are two main routing mechanisms can be used over pub/sub systems, either broker-based or broker-less (Peer-to-Peer). The broker-based methods are used by default in TinyDDS to route its messages of subscription and publication. The centralized method (broker-based) such as DefTDDS causes a bottleneck problem when it is used in WSAN, which makes it not appropriate for such platform. This problem exhausts the node energy rapidly and ends the life time of the network that still has enough residual energy. However, this problem can be mitigated by using the other approaches, BLTDDS and HyTDDS. Thus, in this section we present and study three approaches of pub/sub system: DefTDDS, BLTDDS and HyTDDS.

3.2.1 Default TinyDDS

In DefTDDS, One Rendezvous Node (RN/Broker) is assigned with each topic. The RN acts as a meeting point for publishers and subscribers interested in a particular topic. The DHT-based event protocol is used to apply the hash function that maps the topic to a sensor node ID. This node is called the hashed node which is the broker node in the case of DefTDDS. Therefore, if a node needs to subscribe to a topic, the node information and the subscription related to that topic are sent in the Discovery phase to the hashed node. The end nodes are able to obtain the RN address depending on the hashing function shown in Equation 3.1. This equation shows the identification of the Topic and maximum Topic numbers which are known before deploying a network since they are part of parameters of the middleware core.

$$RN_{address} = Topic_{id} \text{ Modulus MaxTopics} \qquad (3.1)$$

Similarly, when a node publishes an event, this event is transmitted to the hashed node related to the topic of the event. Thus, in the Data Dissemination phase, the information that has been preserved by the hashed node can be used to forward the event to the subscribers. The hashed node (RN) memory does not save the new publication, it forwards it immediately to the interested subscribers registered in the hashed node database. After that, this new publication will be deleted and so on. If there are two or more subscribers for a single topic, the hashed node only receives single publication and then multicasts it to all interested subscribers. Figure 28 shows both the discovery and data dissemination phases of DefTDDS approach.

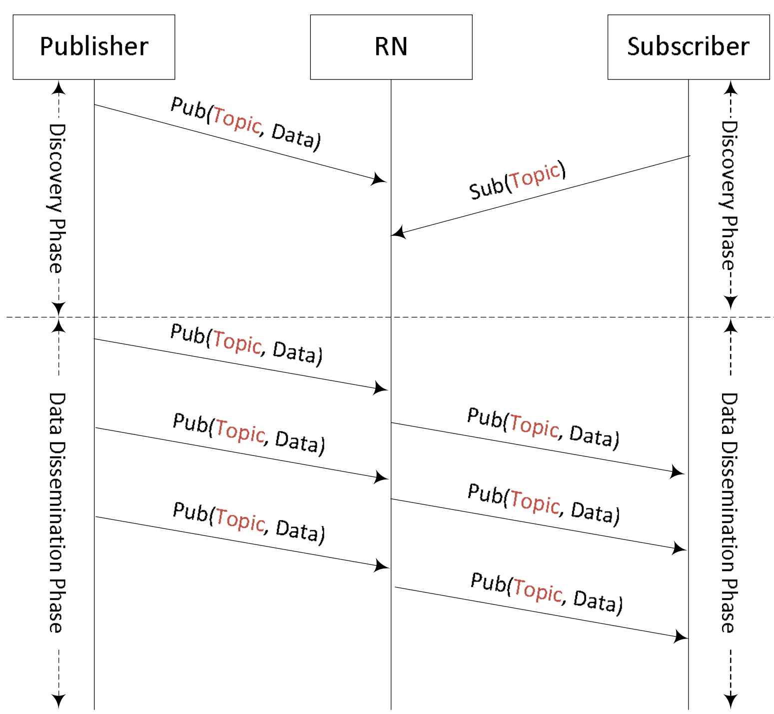

Figure 28. Discovery phase and Data dissemination phase in DefTDDS

3.2.2 Broker-Less TinyDDS

In BLTDDS [21], the subscriber broadcasts subscription messages to all nodes in the network. The publisher is responsible for the matching process. BLTDDS eliminates the RN totally by distributing the middleware functionality over the nodes (publishers/subscribers). As depicted in Figure 29, the subscriber in the discovery phase starts broadcasting the node information and its subscription to all hashed nodes (publishers) in the network. Afterward, in the Data Dissemination phase, the information preserved by the publishers can decide whether to send its data to the subscriber or not. If there is a match in the first phase, then the data dissemination phase can start, and each publisher which has the interested topic can start sending data to the corresponding or interested subscriber. Since this approach eliminates the RN node, it is able to overcome the bottleneck problems and the single point of failure. Furthermore, BLTDDS can distribute the network load over all nodes deployed in that network efficiently and in a more effective way compared to DefTDDS approach. Nevertheless, the number of subscription messages increments highly under heavy network load because it is a flooding-based approach that leads to the next approach.

3.2.3 Hybrid TinyDDS

This approach has been proposed by [62] to minimize the flooding overhead in BLTDDS and mitigate the effect of bottleneck problem in DefTDDS. It uses the RN only in the discovery phase and eliminates it in the data dissemination phase. The hashed node/RN is responsible for mapping the node ID to the topic. In the discovery phase, the subscription messages are forwarded by the hashed node to the matched publishers. Then the hashed node uses the notification messages to communicate with the corresponding publishers. Thus, the list of subscribers is distributed over all publishers where each publisher maintains its all interested subscribers. This approach is shown in Figure 30 where the data is sent immediately to the interested subscribers after the discovery phase.

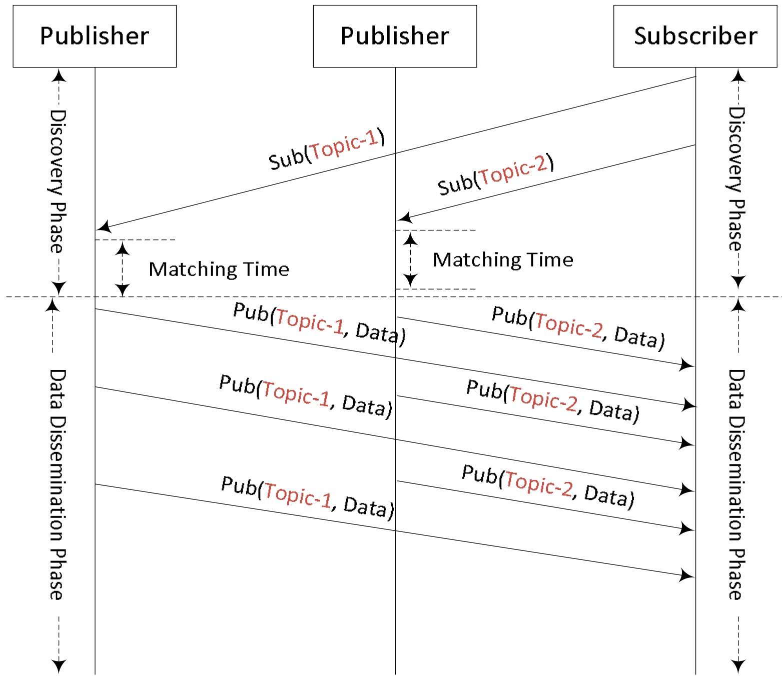

Figure 29. Discovery phase and Data dissemination phase in BLTDDS

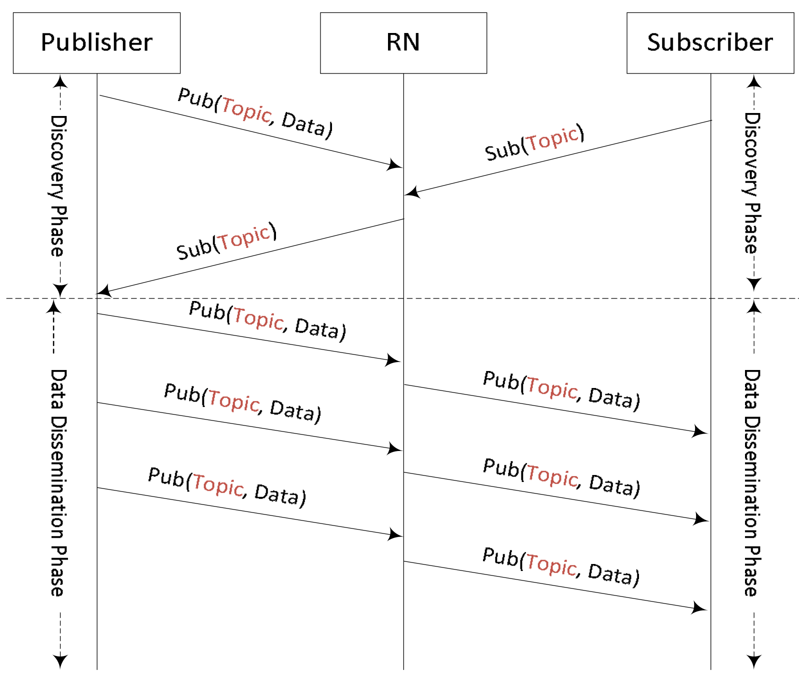

Figure 30. Discovery phase and data Dissemination phase in HyTDDS

3.3 Performance Evaluation

In this section, we present performance evaluation of three TinyDDS platform approaches (DefTDDS, BLTDDS and HyTDDS) using TOSSIM network simulator. The experiments were setup to evaluate the performance of these three approaches under varying network load (2, 4, 6, and 8 packets/second), using the following metrics.

- ***Throughput:*** This metric measures the number of units of information that a system can deliver in a given amount of time. It is used widely in various systems and different aspects of network systems and computers. Throughput can be defined as.

$$Throughput = \frac{Number\ of\ deleivered\ Packet * Packet\ size}{Total\ Duration\ of\ Simulation} \qquad (3.2)$$

- ***Packet Delivery Ratio (PDR):*** It is defined as the ratio of successfully received packets by a destination to the number of packets which have been sent by the source. We can calculate PDR by using the following expression.

$$PDR = \frac{Number\ of\ deleivered\ packet\ * 100}{Number\ of\ sent\ packet}\ \% \qquad (3.3)$$

- ***End-to-End Delay***: This metric denotes the time required to transmit a message across a network from the source/publisher to the destination/subscriber. It is a popular term used for monitoring in an IP network.
- ***Energy Consumption:*** This metric provides an idea about the energy efficiency of any proposed approach in a specific round. It measures the total amount of energy (in milli-Joule) spent in transmitting messages during a particular period of time and can determine the network lifetime.

3.3.1 Simulation Setup

Table 3 shows the setup parameters used to conduct the comparative study of DefTDDS, BLTDDS and HyTDDS approaches. Several experiments were conducted to evaluate the performance of the three approaches. Figure 31 depicts the main scenarios used in simulation experiments. In our evaluation, we use a grid topology because this topology is used in the default TinyDDS (4 x 4 grid) involved in this study.

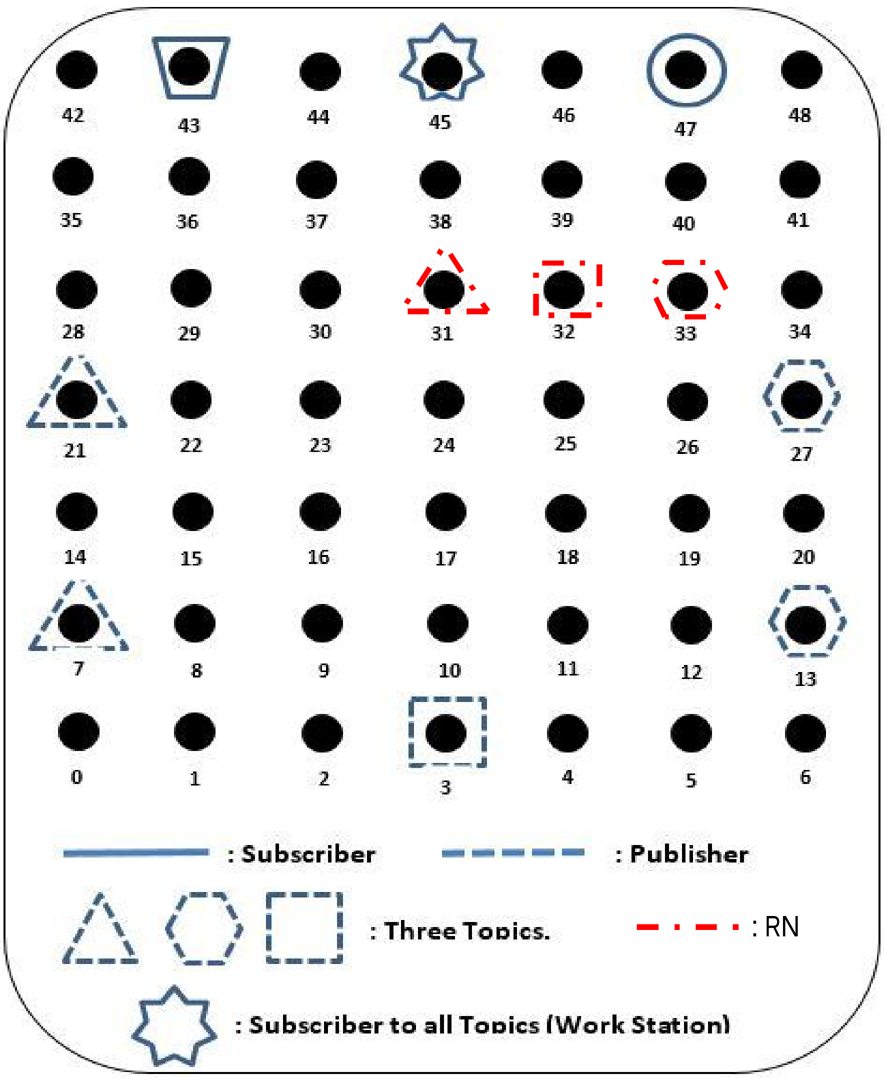

Figure 31. The main scenario with 5 publishers, 3 subscribers and 3 topics

Table 3 Setup parameters

Parameter	Value
Topology	Squared grid
Area	100 $Meter^2$
Number of Nodes	49
Simulation time	1000 seconds
Mote platform	MicaZ
Data rates	2, 4, 6, 8 messages/second
Message size	20 bytes
Number of publishers	5
Number of subscribers	3
Number of topics	3
Runs per results' data point	10

3.3.2 Results and analysis

Throughput:

Figure 32 and Table 4 show the effect of the network load (i.e. number of packets sent per second) on each approach in terms of throughput. It is obvious that BLTDDS improves the network throughput and performs the best performance. That is because each publisher is responsible for the matching process. Afterward, the broadcasting mechanism is used in both two phases to serve the subscribers. The HyTDDS approach is still performing better than the DefTDDS since it uses the RN only to process the discovery phase then it uses the broadcasting technique in the data dissemination phase to respond to the subscribers that has been listed on the publishers in the first phase. Intuitively, the network throughput is proportional to the packet rate in HyTDDS, BLTDDS. DefTDDS shows the worst case especially under high traffic rate (6, 8 packets/second). The reason behind that, this approach relies on the RN to do the hashing and to disseminate data. In DefTDDS, there is only one way/path to serve the discovery and data dissemination phases which is through the RN. As a result, the impact of the bottleneck problem appears and causes high drop and retransmission rate of the packets.

Table 4. Raw Data of Throughput (bytes/second)

Packets/second	2	4	6	8
BLTDDS	60.78	71.76	82.06	93.34
HyTDDS	49.392	65.024	76.9	87.784
DefTDDS	24.472	49.664	62.316	66.112

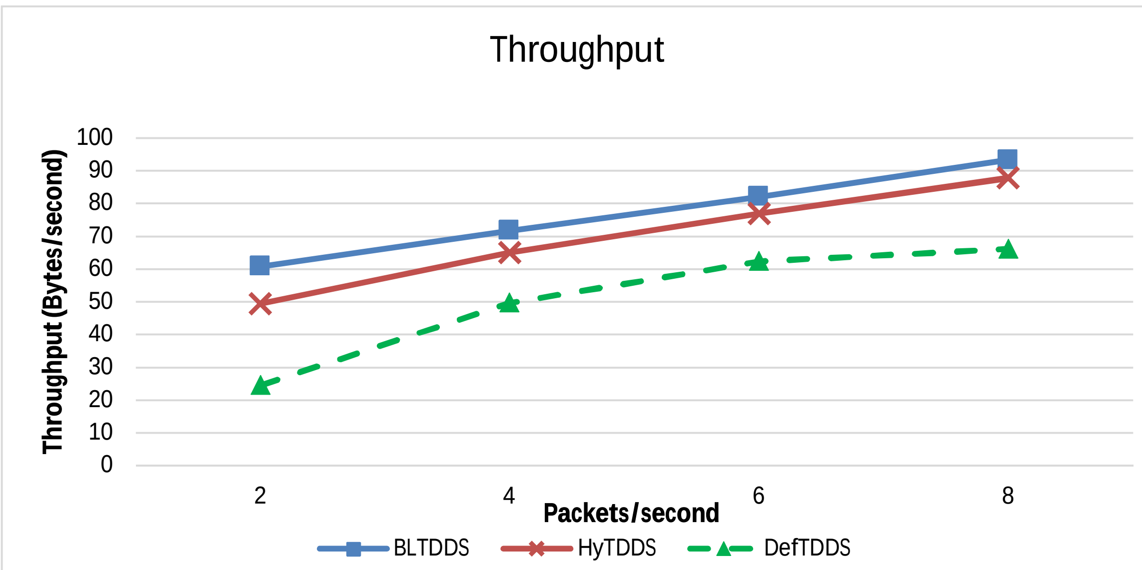

Figure 32. Throughput of DefTDDS, BLTDDS and HyTDDS

Packet Delivery Ratio:

In general, higher throughput gives higher PDR and vice versa. The PDR performance evaluation is depicted in Figure 33. Both BLTDDS and HyTDDS allow subscribers and publishers to communicate directly in the data dissemination phase (i.e. using the broadcasting mechanism) which can guarantee a high percentage of delivered packets. Thus, the two approaches can perform better performance in terms of PDR as shown in Table 5. Moreover, the PDR increases in these two approaches at heavy network load since they can avoid the bottleneck problem in the second phase. In case of DefTDDS, there is only one way to discover a topic and its interested subscribers and to disseminate data which is through the hashed node (RN). Consequently, the PDR in DefTDDS is instable and as long as the packet rate increases, the more dropped packets occurs.

Table 5. Raw Data of PDR (%)

Packets/second	2	4	6	8
BLTDDS	99.969	99.965	99.969	99.972
HyTDDS	99.925	99.963	99.973	99.949
DefTDDS	98.244	98.214	97.902	90.432

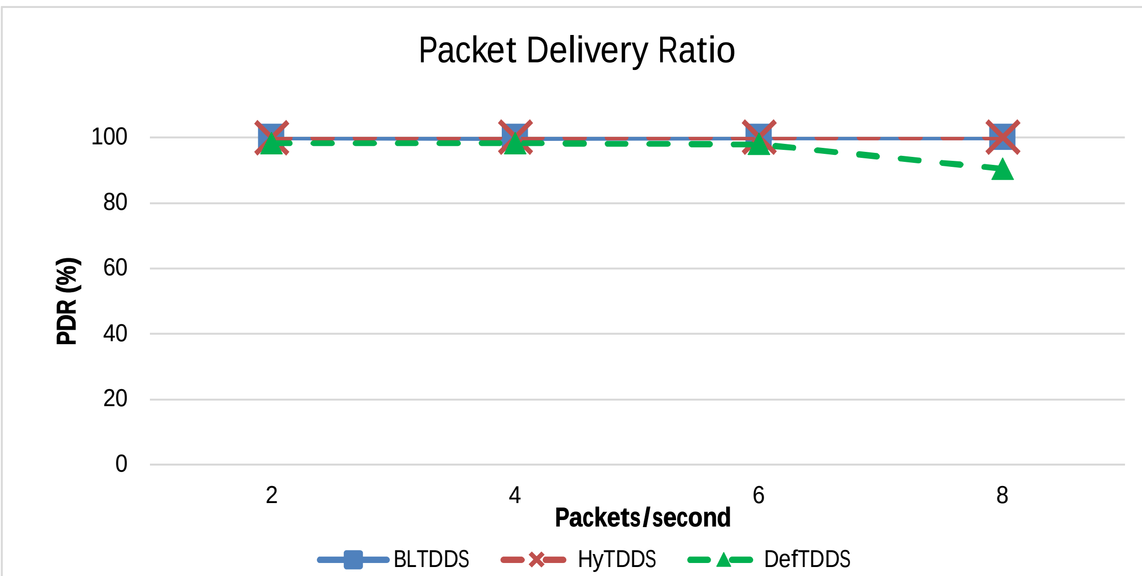

Figure 33. Packet Delivery Ratio of DefTDDS, BLTDDS and HyTDDS

EED performance evaluation metric is depicted in Figure 34 and Table 6 where BLTDDS shows the best performance. In BLTDDS, a subscriber broadcasts the subscription messages to all nodes in the discovery phase where each publisher is responsible for the matching process. Because of that, BLTDDS can perform a stable EED and speed up the process of exchanging messages in both two phases. The DefTDDS approach uses the RN to implement the hash function, to do the matching process and then to serve subscriptions and publications in the dissemination phase. That leads to slow exchanging messages between publishers and subscribers. In case of HyTDDS, although this approach uses the broadcasting mechanism in the data dissemination phase, it still shows high delay because of the long process done in the discovery phase. First, it uses the RN for hashing and then each publisher has to maintain its all interested subscribers using the notification messages between the RN and the publishers. By the end of the discovery phase, each publisher should have a list of all interested subscribers to start the second phase. Generally, BLTDDS is still the best option for real-time applications in WSAN networks in terms of EED and throughput.

Table 6. Raw Data of EED (ms)

Packets/second	2	4	6	8
BLTDDS	14.291	15.279	15.544	15.582
HyTDDS	44.278	44.657	44.776	46.253
DefTDDS	43.429	44.483	44.485	45.435

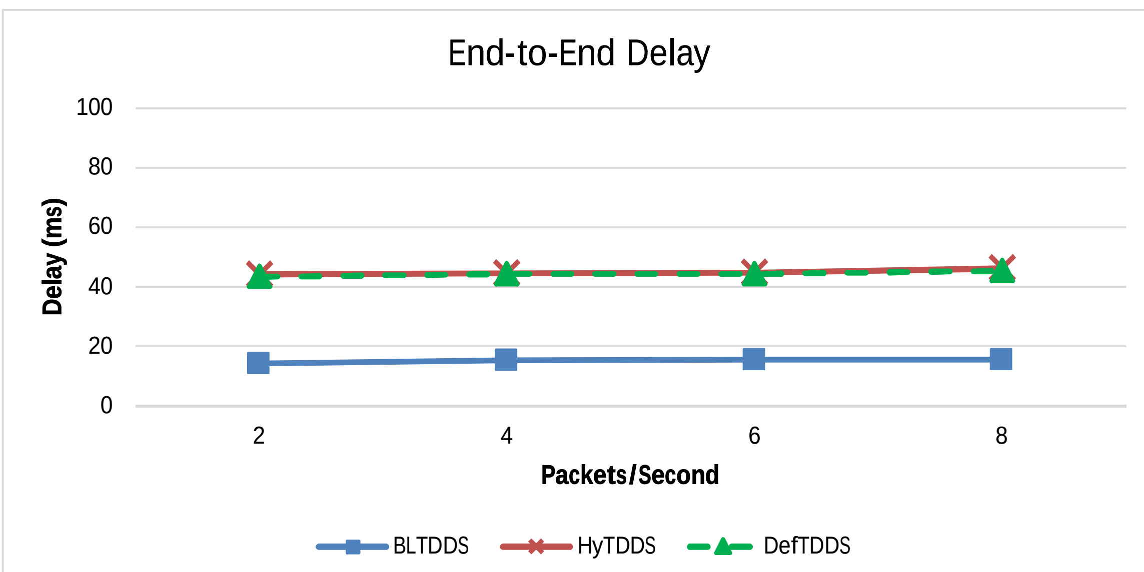

Figure 34. End-to-End Delay of DefTDDS, BLTDDS and HyTDDS

Energy Consumption:

In this section, we consider the Radio and MCU components in our evaluation since they have the most effect on the energy consumption metric, specifically the Radio component. We study and analyse the results of energy consumption of DefTDDS, BLTDDS and HyTDDS approaches over these two components. We use the MicaZ Energy model parameters used in the study [21], Table 7 shows the parameters of MicaZ energy model.

Table 7. MicaZ Energy model parameters

MCU		**Radio**	
Mode	**Current**	**Mode**	**Current**
Active	8 mA	Receiving	19.7 mA
Idle	4 mA	Transmitting	17.4 mA
Sleep	9 uA	Sleep	1 uA

We have evaluated the energy consumption under various network load as energy consumption of two main components, Radio and MCU. The results of this evaluation are shown in Figure 35 and Figure 36 and provided as raw data in Table 8 and Table 9. Energy consumption in both two components shows that the BLTDDS outperform the HyTDDS and DefTDDS, especially under low traffic rate (2, 4 packets/second). The reason behind that, the number of retransmission is very low since there is multiple path for the pub/sub communications. In contrast, the DefTDDS approach shows the worst case in terms of energy consumption because it enforces all subscribers and publishers to communicate through the RN even if they are close to each other than the RN. Thus, the messages among subscribers and publishers might need to take long way of transmission which leads to consume more energy.

In case of HyTDDS, this approach performs the best performance under heavy network load (6, 8 packets/second) due to its ability to control the radio and MCU energy consumption. This approach reduces the impact of the flooding overhead by taking the advantage of centralized systems, and decentralized systems in the first phase and second phase, respectively. HyTDDS uses the broadcasting mechanism only in the second phase which reduce the radio energy consumption. Moreover, it reduces the MCU energy consumption since the most of the MCU time is in the idle or sleep mode. In general, the MCU energy consumption can be neglected since it is very low compared to the radio energy consumption. Therefore, The HyTDDS is considered the more stable and efficient approach in terms of energy consumption under different traffic rate.

Table 8. Raw Data of REC Energy Consumption in Milli-joule

Packets/second	2	4	6	8
BLTDDS	7113.226	10251.878	13006.06	15089.88
HyTDDS	10607.44	11271.92	11194.16	11426.14
DefTDDS	14324.32	14363.72	14391.38	14408.82

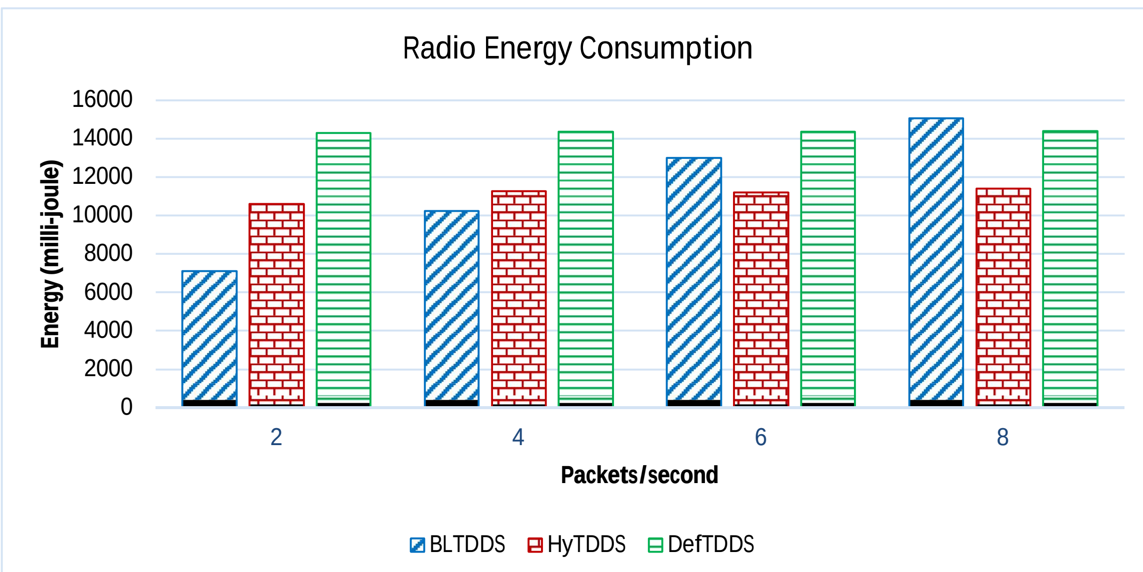

Figure 35. Radio Energy Consumption of DefTDDS, BLTDDS and HyTDDS

Table 9. Raw Data of MCU Energy Consumption in Milli-joule

Packets/second	2	4	6	8
BLTDDS	327.5866	645.8124	956.4782	1204.436
HyTDDS	618.746	660.4422	725.1976	752.2614
DefTDDS	1105.226	1210.528	1310.124	1414.986

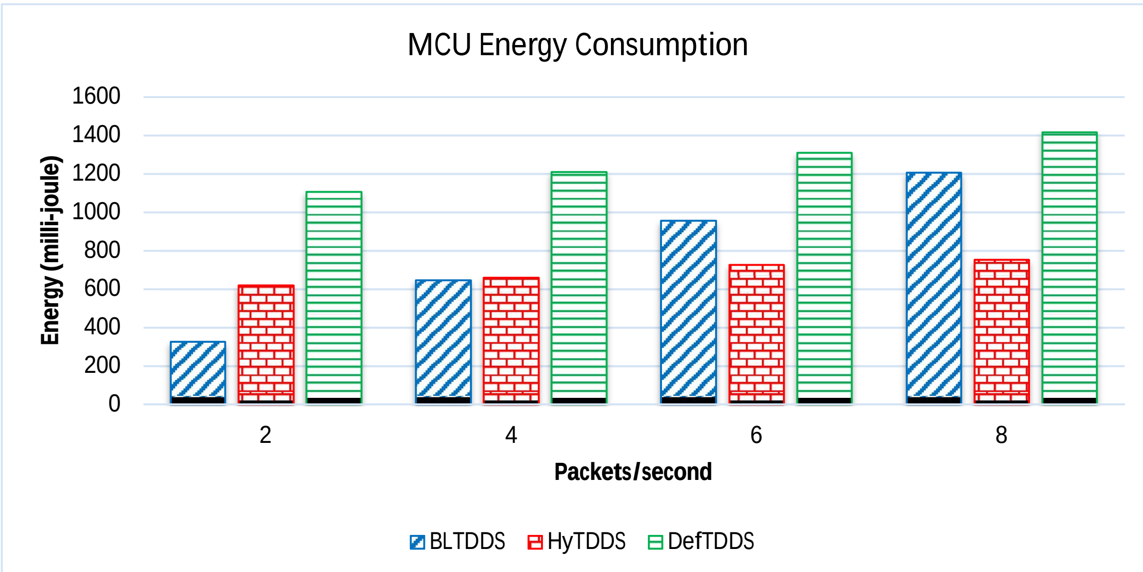

Figure 36. MCU Energy Consumption of DefTDDS, BLTDDS and HyTDDS

3.3.3 TinyDDS Approaches Classifications

In WSAN, it is not easy to provide a certain pub/sub communication system that can deal with all applications requirements. This is because the performance of these approaches relies on different factors such as the traffic rate, how data is disseminated, application requirements in terms of throughput, delay, and energy consumption. In this chapter, we studied, evaluated and analysed the three TDDS approaches under different traffic rate. According to our findings and the parameters discussed in chapter 1, although the three approaches are appropriate for real-time applications, we still consider BLTDDS is the best choice for such applications. Moreover, BLTDDS and HyTDDS can meet the requirements of High-rate traffic applications. For the harsh environments where changing the batteries is very difficult, we recommend using HyTDDS approach for High-rate traffic applications and BLTDDS for Medium-rate and Low-rate traffic applications.

CHAPTER 4

Enhanced Energy Aware TinyDDS

In WSANs, the battery (i.e. AA batteries that are a common choice in such networks) is the main source of power in such networks. It is required to supply the energy required to achieve a programmed task on the S/A devices. Batteries have limited energy and it may be inconvenient to be recharged, especially when S/A devices are used in a hostile environment. On the other side, these devices should have enough energy to fulfill the requirements of an application. Hence, energy is a resource in WSANs and should be utilized carefully. Therefore, the energy optimization is an important factor for the design of sensor-based networks. In this chapter, we add improvements to the EATDDS protocol proposed in [21]. We use TinyDDS middleware to present, implement and evaluate the enhanced pub/sub energy aware protocol.

4.1 E-EATDDS Protocol

Since the TinyDDS implementation uses a grid topology as an examined topology, we implement the scenario of the proposed E-EATDDS approach using the same topology. In this study, the energy consumption of each node in the network is monitored by Online Energy Model (OEM) discussed in chapter 2. OEM helps nodes to monitor their energy level and then send the energy information to the RN periodically. Figure 37 shows how the discovery phase and data dissemination phase can be performed in E-EATDDS. In E-EATDDS, there is only one broker works at the same time and hosts one topic, assuming we have only one topic and each node knows the locations of other nodes. The RN checks its energy level periodically. When the RN node reaches a predefined threshold, it selects a new RN that has the maximum remaining energy. Then the new selected RN node broadcasts the ACK_Message to all nodes in the network. In Figure 37, we assume that the RN has selected the Successor RN (SRN) which has the maximum remaining energy. The main difference between EATDDS and the proposed E-EATDDS is that, EATDDS protocol depends on rounds (time) to elect the RN node while in E-EATDDS the new RN node will be elected based on a predefined threshold (i.e. specific energy level).

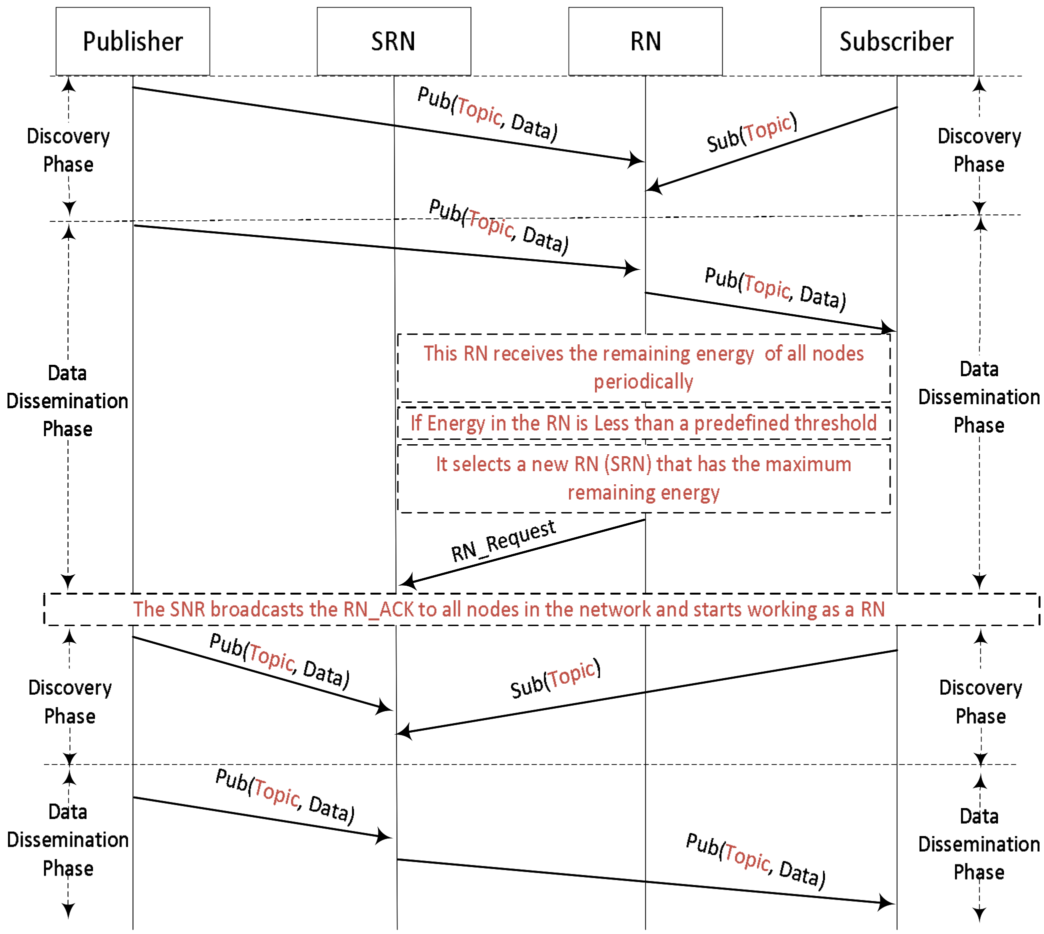

Figure 37. Discovery phase and data Dissemination phase in E-EATDDS

E-EATDDS is a cluster-based network as same as EATDDS protocol that was inspired by the LEACH-C protocol proposed in [64]. Since we use the same setup parameters discussed in chapter

3 to evaluate E-EATDDS, we have three clusters, one for each topic. Each cluster contains all the publishers and subscribers which are relevant to that topic. Meaning that, there are three RN nodes that are responsible for hosting the three topics and managing three clusters. Each topic is mapped to a RN address as discussed earlier. If there is more one RN node, the energy information of each node will be sent to all RN nodes in the network. Figure 38 shows the RN node switching process.

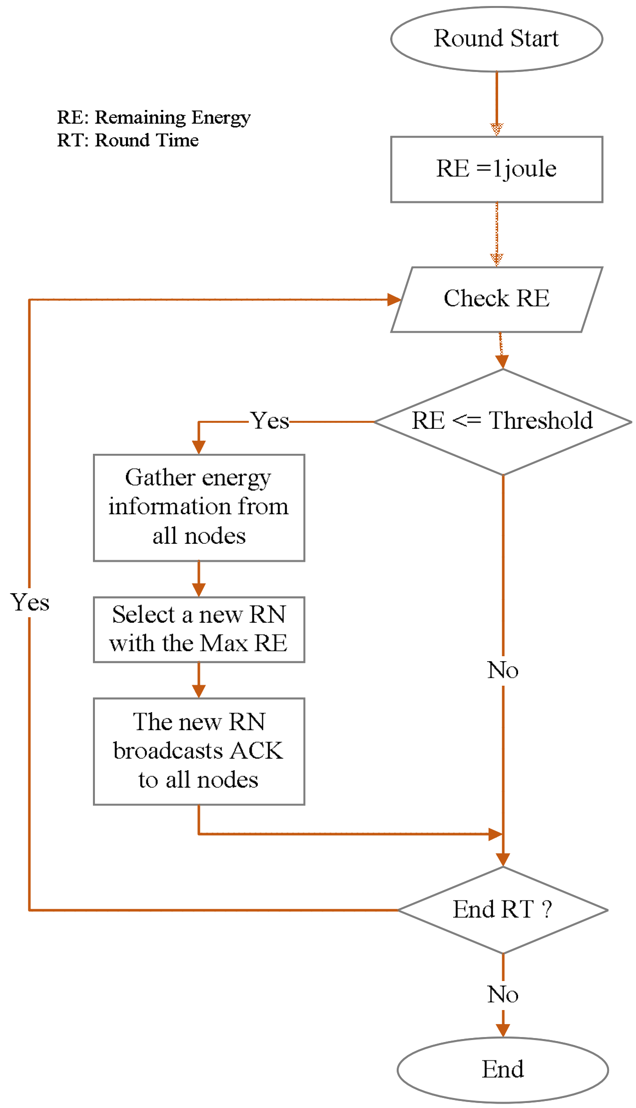

Figure 38. Switching Algorithm Flowchart

4.2 Performance Evaluation

In this section, E-EATDDS is tested and evaluated under different traffic loads. Since we have added enhancements to an energy aware protocol (EATDDS), we concentrate on the energy consumption metric. There are other metrics that reflect the energy consumption over the network such as network life time and number of packets sent per joule. In the previous comparative study conducted in chapter 3, we used a limited time (1000 ms) to evaluate the three TinyDDS approaches in chapter 3. In E-EATDDS, when the first node dies, the measurements of our experiments are taken regardless of the simulation time. Figure 39 shows the tested scenario as a cluster-based network where each RN is responsible for one cluster.

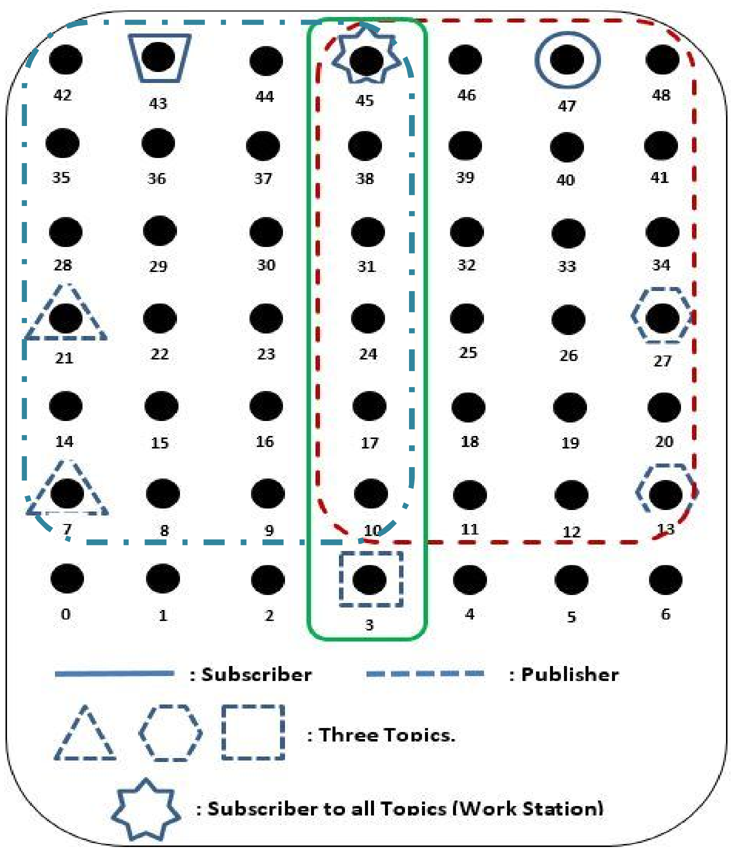

Figure 39. Cluster Formation of The Tested Scenario

4.2.1 Experiment setup

We use the same scenario and simulation parameters set up used in the comparative study, chapter 3. The only difference is that we use unlimited time in order to measure the network life time at the moment the first node dies. The initial energy parameter is one of the most important parameters used in this study. All the nodes in the network would start with the same initial energy and a node would be considered dead once it consumes its whole energy. The initial energy value has been selected to be one joule as presented in previous studies [21, 65]. We assume that the round time of E-EATDDS is 100 seconds. Meaning that every 100 seconds, the RN checks its energy level along with the energy information that have been received from other nodes in the network. When the old RN reaches predefined energy threshold, it selects a new RN that has the maximum remaining energy. Moreover, we use a different traffic rate for all protocols. E-EATDDS is tested under different levels of energy thresholds, which means when a node reaches this amount of energy, a new RN will be selected by the previous RN node to balance the distributed RN nodes.

4.2.2 Performance metrics

We highlight the efficiency and effectiveness of the middleware in terms of energy consumption. Moreover, the performance evaluation of the protocol is tested based on the number of packets received successfully per joule.

Total Remaining Energy: This metric is used to reflect the efficiency of the energy consumption distribution over the nodes in the network. Therefore, a large amount of remaining energy refers to a good mechanism in terms of energy savings. It can be measured by getting the summation of the remaining energy of all nodes in the network. The measurements of energy are in milli-Joule.

Packet per Joule: It is a good measurement metric used to reflect the protocol efficiency in terms of energy-saving. It is defined as the number of delivered packets divided by the total energy consumption during the network life time.

Network Life Time: In this chapter, the network life time is defined as the duration of time till the first node in the network dies. It mainly relies on the life time of the battery and a node dies when it exhausts its energy. The initial energy of each node is one joule.

4.2.3 Results and analysis

First, we assumed that, we have three energy levels as shown in Figure 40. We tested and evaluated the proposed E-EATDDS under these three different thresholds. Figure 41 depicts the effect of energy levels on Packet per joule metric. We notice that E-EATDDS shows better performance at low threshold because the network will exploit the time for exchanging messages rather than electing a new RN every small amount of time. Thus, we select the threshold to be 25% of the maximum energy, in our case the threshold is 250 milli-joule.

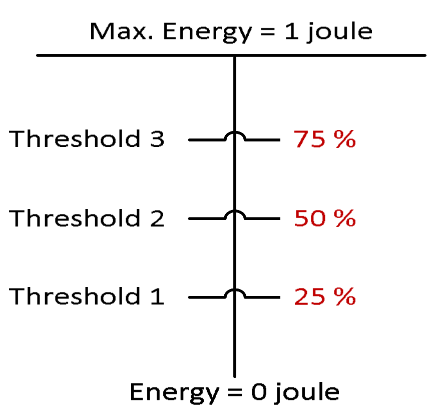

Figure 40. Three Energy Levels of The RN

Figure 42 depicts the total remaining energy of the three protocols discussed in chapter 3 beside EATDDS and E-EATDDS protocols. The more remaining energy the more efficient usage of energy. The DefTDDS shows the minimal total remaining energy, it performs the most wasted energy. Meaning that, DefTDDS protocol has done less work from the time the network starts its operation until it is over. As broker-based protocols, the EATDDS and E-EATDDS protocols provide the most effective performance and give the longest life time compared to the DefTDDS. E-EATDDS protocol performs slightly better than EATDDS since this protocol depends directly on the energy information in the RN election process.

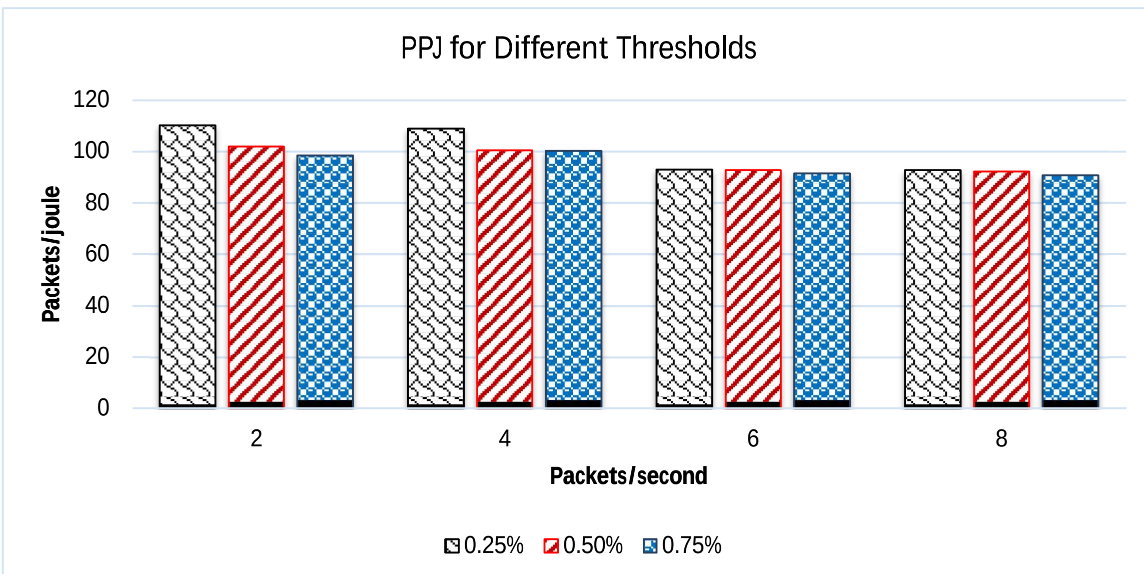

Figure 41. Packets per Joule Under Different Energy Level

The packet per joule is a prominent metric used by many studies to show the energy efficiency. The more received packets per joule is the better. Figure 43, E-EATDDS protocol outperforms EATDDS protocol and the other three protocols in case of low network load. As an energy aware protocols, EATDDS and E-EATDDS have almost the same performance under heavy network load. E-EATDDS shows a degradation in performance in terms of packet per joule during the high network load because the RN nodes deplete their energy much faster which leads to high rate of

RN election and then less delivered packets. Generally, E-EATDDS is the best protocol in terms of packet per joule metric.

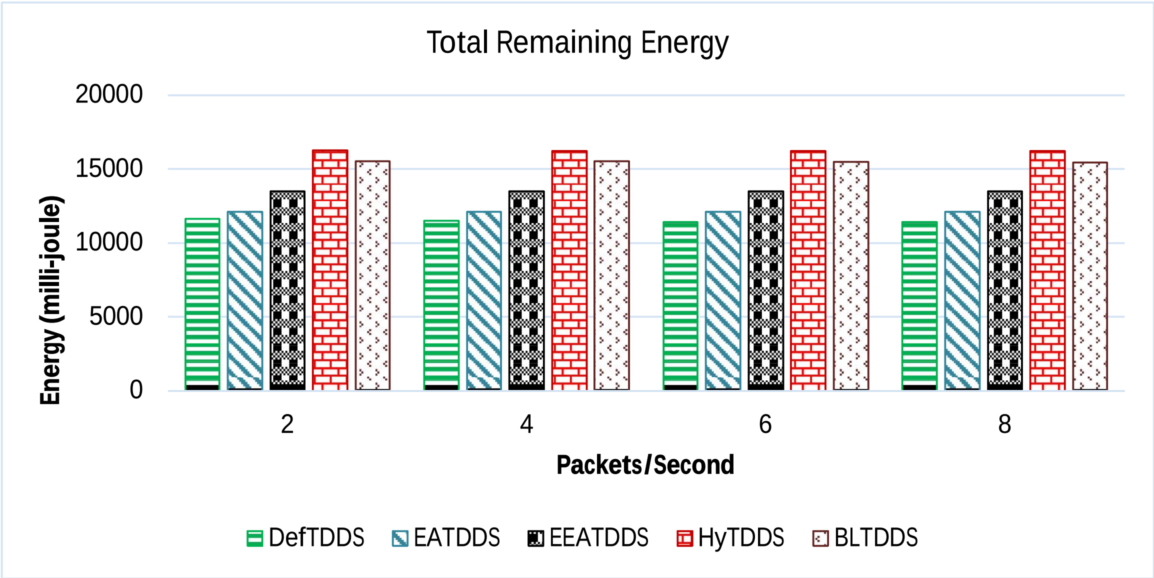

Figure 42. Total Remaining Energy at The Moment The First Node Dies

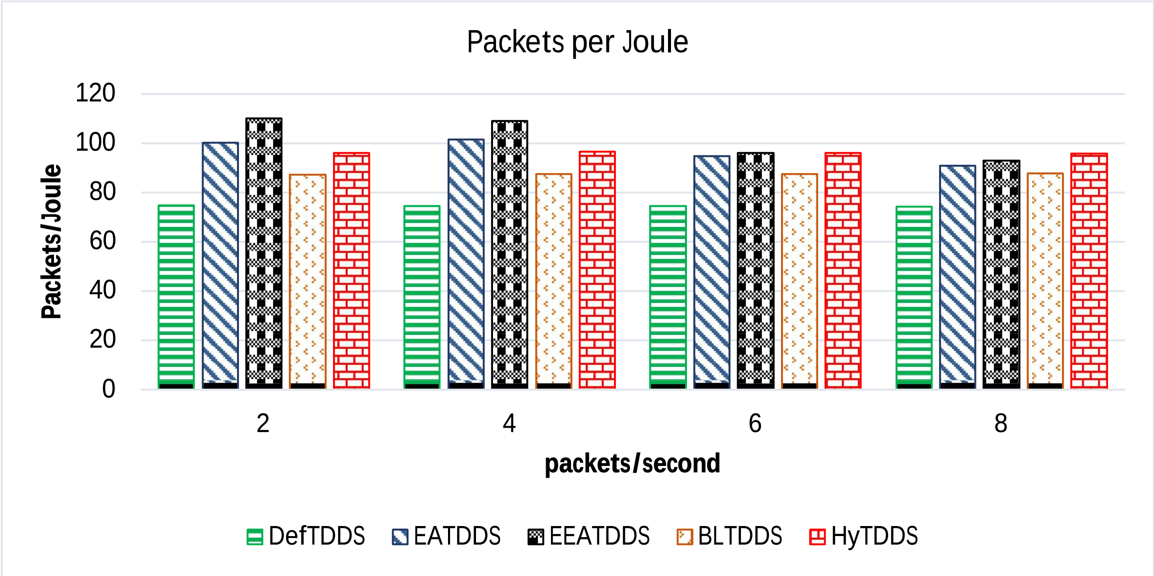

Figure 43. Packets per Joule under Different Network Load

Figure 44 presents the effect of each protocol on the network life time metric. BLTDDS and HyTDDS show almost the same and highest performance in terms of network life time under light and heavy network load since they differ only in the discovery phase process. HyTDDS outperforms all protocols at heavy network load. Although BLTDDS and HyTDDS outperform the two energy protocols (EATDDS and E-EATDDS) in terms of network life time, they still waste part of energy because they are flooding-based approaches. Consequently, the energy aware protocols can use the network life time efficiently. In E-EATDDS, the new RN is elected based on the energy level information on each node, unlike the EATDDS that selects a new RN every round. In EATDDS, the load on the RN might be too heavy in some rounds that would end the network life time before starting the next round.

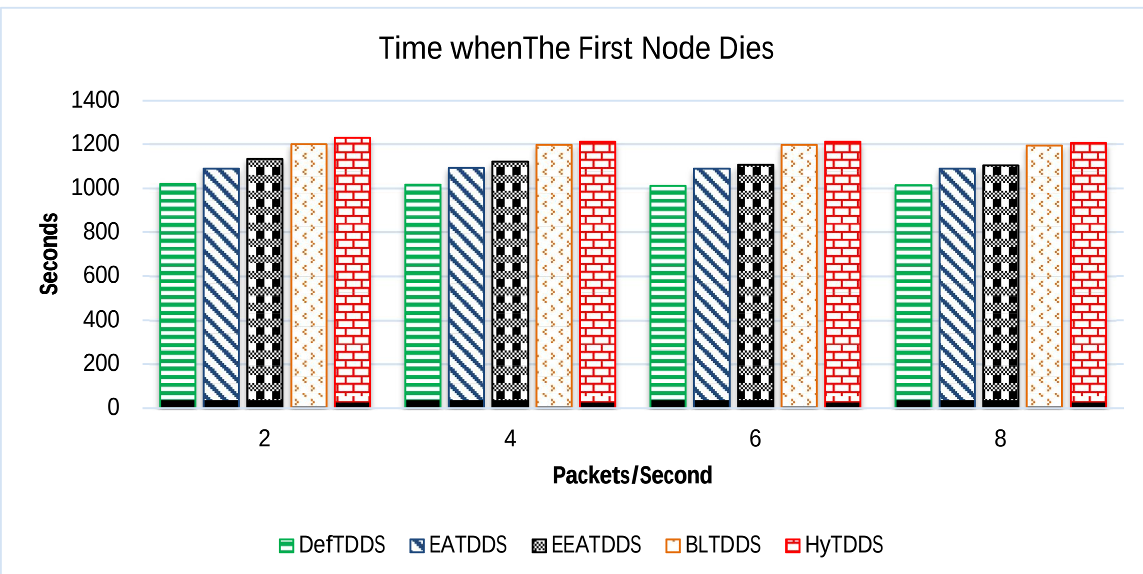

Figure 44. Time at The Moment The First Node Dies

CHAPTER 5

CONCLUSIONS AND FUTURE WORKS

In this thesis, we presented the pub/sub interaction system over WSAN. We reviewed its architecture, components, properties and limitations. Moreover, a survey about the recent pub/sub middleware solutions in the context of WSAN has been conducted. According to the pub/sub solutions in the literature, pub/sub interaction systems that have limited network resources require more improvements in terms of different aspects such as implementation and design. There are many improvements can be conducted over resources-constraints by considering the generality of the middleware the specification degree of an application.

In WSAN, it is not easy to provide a certain pub/sub communication system that can deal with all applications requirements. This is because the performance of these approaches relies on different factors such as the traffic rate, data dissemination, delay, and energy consumption. In this work, we studied, evaluated and analysed three TDDS approaches under different traffic rates. According to our findings and the parameters discussed in chapter 3 and chapter 1, respectively, BLTDDS is the best choice out of three protocols for real-time applications. Moreover, BLTDDS and HyTDDS can meet the requirements of High-rate traffic applications. For the harsh environments where changing the batteries is very difficult, we recommend using HyTDDS approach for High-rate traffic applications and BLTDDS for Medium-rate and Low-rate traffic applications.

E-EATDDS protocol has improved the EATDDS efficiency in terms of energy consumption. However, the middleware still needs several enhancements to be used widely. In E-EATDDS, when a RN node reaches a predefined threshold of energy, it selects a new RN that has the maximum remaining energy from its cluster. We set the threshold to be 25% of the maximum energy of the node because of the high performance that can be achieved at this level of energy. The data dissemination techniques can be used to reduce the total energy consumption and End-to-End delay.

We used the grid topology in this study since the default TinyDDS is involved in our comparative study and this topology is also used commonly in different indoor and outdoor applications. The distribution of energy consumption depends on underlying routing protocols affected by the topology. Thus, using a probabilistic topology to evaluate an energy aware protocol such as E-EATDDS might lead to new performance and implementation issues. As a future work, different network topologies (e.g. uniform and random topology) might be used to evaluate E-EATDDS protocol. Moreover, E-EATDDS has different parameters that can be analysed and improved such as synchronization, and the clustering technique.

References

[1] L. Krishnamachari, D. Estrin, and S. Wicker, "The impact of data aggregation in wireless sensor networks," in *Distributed Computing Systems Workshops, 2002. Proceedings. 22nd International Conference on*, 2002, pp. 575-578.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer networks,* vol. 38, pp. 393-422, 2002.

[3] M. L. McKelvin Jr, M. L. Williams, and N. M. Berry, "Integrated radio frequency identification and wireless sensor network architecture for automated inventory management and tracking applications," in *Proceedings of the 2005 Conference on Diversity in Computing*, 2005, pp. 44-47.

[4] T. Sheltami, A. Al-Roubaiey, A. Mahmoud, and E. Shakshuki, "A publish/subscribe middleware cost in wireless sensor networks: A review and case study," in *Electrical and Computer Engineering (CCECE), 2015 IEEE 28th Canadian Conference on*, 2015, pp. 1356-1363.

[5] S. Oh, J.-H. Kim, and G. Fox, "Real-time performance analysis for publish/subscribe systems," *Future Generation Computer Systems,* vol. 26, pp. 318-323, 2010.

[6] C. Esposito, D. Cotroneo, and S. Russo, "Survey On reliability in publish/subscribe services," *Computer Networks: The International Journal of Computer and Telecommunications Networking,* vol. 57, pp. 1318-1343, 2013.

[7] R. Baldoni, M. Contenti, and A. Virgillito, "The evolution of publish/subscribe communication systems," in *Future directions in distributed computing*, ed: Springer, 2003, pp. 137-141.

[8] A. R. Ribeiro, F. Silva, L. C. Freitas, J. C. Costa, and C. R. Francês, "SensorBus: a middleware model for wireless sensor networks," in *Proceedings of the 3rd international IFIP/ACM Latin American conference on Networking*, 2005, pp. 1-9.

[9] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," *ACM computing surveys (CSUR),* vol. 35, pp. 114-131, 2003.

[10] U. Hunkeler, H. L. Truong, and A. Stanford-Clark, "MQTT-S—A publish/subscribe protocol for Wireless Sensor Networks," in *Communication systems software and middleware and workshops, 2008. comsware 2008. 3rd international conference on*, 2008, pp. 791-798.

[11] D. Chen and P. K. Varshney, "QoS Support in Wireless Sensor Networks: A Survey," in *International conference on wireless networks*, 2004, pp. 1-7.

[12] M. Sharifi, M. A. Taleghan, and A. Taherkordi, "A middleware layer mechanism for QoS support in wireless sensor networks," in *Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies, 2006. ICN/ICONS/MCL 2006. International Conference on*, 2006, pp. 118-118.

[13] F. Xia, "QoS challenges and opportunities in wireless sensor/actuator networks," *Sensors,* vol. 8, pp. 1099-1110, 2008.

[14] J. Ferreira and F. J. Velez, "Enhanced UMTS services and applications characterisation," *Telektronikk,* vol. 101, p. 113, 2005.

[15] F. J. Velez and L. M. Correia, "Mobile broadband services: classification, characterization, and deployment scenarios," *IEEE Communications Magazine,* vol. 40, pp. 142-150, 2002.

[16] L. M. Borges, F. J. Velez, and A. S. Lebres, "Survey on the characterization and classification of wireless sensor network applications," *IEEE Communications Surveys & Tutorials,* vol. 16, pp. 1860-1890, 2014.

[17] Y. Chen, T. Farley, and N. Ye, "QoS requirements of network applications on the Internet," *Information Knowledge Systems Management,* vol. 4, pp. 55-76, 2004.

[18] B. Krishnamachari, D. Estrin, and S. Wicker, "Modelling data-centric routing in wireless sensor networks," in *IEEE infocom*, 2002, pp. 39-44.
[19] D. A. Tran and L. H. Truong, "Enabling publish/subscribe services in sensor networks," *University of Massachusetts, Boston and IBM Zurich Research Laboratory, Switzerland,* 2010.
[20] X. Tong and E. C. Ngai, "A ubiquitous publish/subscribe platform for wireless sensor networks with mobile mules," in *Distributed Computing in Sensor Systems (DCOSS), 2012 IEEE 8th International Conference on*, 2012, pp. 99-108.
[21] A. AL-Roubaiey, "Energy-Aware Publish/Subscribe DDS-Based Middleware for Wireless Sensor and Actuator Networks," *PhD Dissertation,* 2015.
[22] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking," *IEEE/ACM Transactions on Networking (ToN),* vol. 11, pp. 2-16, 2003.
[23] E. Souto, G. Guimarães, G. Vasconcelos, M. Vieira, N. Rosa, C. Ferraz*, et al.*, "Mires: a publish/subscribe middleware for sensor networks," *Personal and Ubiquitous Computing,* vol. 10, pp. 37-44, 2006.
[24] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo*, et al.*, "TinyOS: An operating system for sensor networks," in *Ambient intelligence*, ed: Springer, 2005, pp. 115-148.
[25] A. Stanford-Clark and U. Hunkeler, "MQ telemetry transport (MQTT)," *Online].* http://mqtt*. org. Accessed September,* vol. 22, p. 2013, 1999.
[26] A. Stanford-Clark and H. Troung, "MQTT for sensor networks (MQTTs) specifications," *IBM, Oct,* 2007.
[27] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," in *Proceedings of the 7th ACM conference on embedded networked sensor systems*, 2009, pp. 1-14.
[28] J.-H. Hauer, V. Handziski, A. Köpke, A. Willig, and A. Wolisz, "A component framework for content-based publish/subscribe in sensor networks," in *Wireless Sensor Networks*, ed: Springer, 2008, pp. 369-385.
[29] J. Chen, M. Díaz, B. Rubio, and J. M. Troya, "PS-QUASAR: A publish/subscribe QoS aware middleware for Wireless Sensor and Actor Networks," *Journal of Systems and Software,* vol. 86, pp. 1650-1662, 2013.
[30] D. A. Tran and C. Pham, "A content-guided publish/subscribe mechanism for sensor networks without location information," *Computer Communications,* vol. 33, pp. 1515-1523, 2010.
[31] D. A. Tran and C. Pham, "Pub-2-sub: A content-based publish/subscribe framework for cooperative p2p networks," in *International Conference on Research in Networking*, 2009, pp. 770-781.
[32] K. Shi, Z. Deng, and X. Qin, "Tinymq: A content-based publish/subscribe middleware for wireless sensor networks," in *Proc. of International Conference on Sensor Technologies and Applications*, 2011, pp. 12-17.
[33] S. Taherian and J. Bacon, "A publish/subscribe protocol for resource-awareness in wireless sensor networks," in *Proceedings of the International Workshop on Localized Algorithms and Protocols for Wireless Sensor Networks (LOCALGOS'07)*, 2007, pp. 27-38.
[34] P. Boonma and J. Suzuki, "TinyDDS: An interoperable and configurable," *Principles and applications of distributed event-based systems,* p. 206, 2010.
[35] J. Chen, M. Díaz, L. Llopis, B. Rubio, and J. M. Troya, "A survey on quality of service support in wireless sensor and actor networks: Requirements and challenges in the context of critical infrastructure protection," *Journal of Network and Computer Applications,* vol. 34, pp. 1225-1239, 2011.
[36] Y. Tekin and O. K. Sahingoz, "A Publish/Subscribe messaging system for wireless sensor networks," in *Digital Information and Communication Technology and its Applications (DICTAP), 2016 Sixth International Conference on*, 2016, pp. 171-176.

[37] A. Awad, R. Germany, and F. Dressler, "Data-centric cooperative storage in wireless sensor network," in *Applied Sciences in Biomedical and Communication Technologies, 2009. ISABEL 2009. 2nd International Symposium on*, 2009, pp. 1-6.

[38] A.-H. Jallad and T. Vladimirova, "Data-centricity in wireless sensor networks," in *Guide to Wireless Sensor Networks*, ed: Springer, 2009, pp. 183-204.

[39] V. Tsetsos, G. Alyfantis, T. Hasiotis, O. Sekkas, and S. Hadjiefthymiades, "Towards commercial wireless sensor networks: business and technology architecture," *Ad Hoc & Sensor Wireless Networks,* vol. 2, pp. 59-80, 2006.

[40] M.-M. Wang, J.-N. Cao, J. Li, and S. K. Dasi, "Middleware for wireless sensor networks: A survey," *Journal of computer science and technology,* vol. 23, pp. 305-326, 2008.

[41] P. Levis and D. Culler, "Maté: A tiny virtual machine for sensor networks," *ACM Sigplan Notices,* vol. 37, pp. 85-95, 2002.

[42] P. A. Levis, D. E. Gay, and D. E. Culler, *Bridging the gap: Programming sensor networks with application specific virtual machines*: Computer Science Division, University of California, 2004.

[43] R. Gummadi, O. Gnawali, and R. Govindan, "Macro-programming wireless sensor networks using Kairos," in *International Conference on Distributed Computing in Sensor Systems*, 2005, pp. 126-140.

[44] R. Lewis, *Advanced Messaging Applications with MSMQ and MQSeries*: Que, 2000.

[45] A. Stanford-Clark and H. L. Truong, "Mqtt for sensor networks (mqtt-sn) protocol specification," *International business machines (IBM) Corporation version,* vol. 1, 2013.

[46] D. Raphaely, M. Chaliha, N. Bhatt, C. Hall, and J. Wilson, "Oracle Streams Advanced Queuing User's Guide, 11g Release 2 (11.2) E11013-03," 2010.

[47] G. Cugola and H. Jacobsen, "Using publish/subscribe middleware for mobile systems," *ACM SIGMOBILE Mobile Computing and Communications Review,* vol. 6, pp. 25-33, 2002.

[48] M. Sun, "Java™ Message Service," *[Online],* 9 November 1999 1999.

[49] OMG, "CORBA notification service specification,," *[Online],* 11 OCT. 2004 2004.

[50] OMG, "CORBA event service specification,," *[Online],* 2004.

[51] M. Castro, P. Druschel, A.-M. Kermarrec, and A. I. Rowstron, "SCRIBE: A large-scale and decentralized application-level multicast infrastructure," *IEEE Journal on Selected Areas in communications,* vol. 20, pp. 1489-1499, 2002.

[52] S. Baehni, P. T. Eugster, and R. Guerraoui, "Data-aware multicast," in *Dependable Systems and Networks, 2004 International Conference on*, 2004, pp. 233-242.

[53] D. P. Anderson, "Metascheduling for continuous media," *ACM Transactions on Computer Systems (TOCS),* vol. 11, pp. 226-252, 1993.

[54] P. Eugster, "Type-based publish/subscribe: Concepts and experiences," *ACM Transactions on Programming Languages and Systems (TOPLAS),* vol. 29, p. 6, 2007.

[55] A. K. Y. Cheung and H.-A. Jacobsen, "Load balancing content-based publish/subscribe systems," *ACM Transactions on Computer Systems (TOCS),* vol. 28, p. 9, 2010.

[56] F. Fabret, H. A. Jacobsen, F. Llirbat, J. Pereira, K. A. Ross, and D. Shasha, "Filtering algorithms and implementation for very fast publish/subscribe systems," in *ACM SIGMOD Record*, 2001, pp. 115-126.

[57] A. Corsaro, L. Querzoni, S. Scipioni, S. T. Piergiovanni, and A. Virgillito, "Quality of service in publish/subscribe middleware," *Global Data Management,* vol. 19, pp. 1-22, 2006.

[58] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf, "Design and evaluation of a wide-area event notification service," *ACM Transactions on Computer Systems (TOCS),* vol. 19, pp. 332-383, 2001.

[59] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: Accurate and scalable simulation of entire TinyOS applications," in *Proceedings of the 1st international conference on Embedded networked sensor systems*, 2003, pp. 126-137.

[60] M. Memsic, "MICAz," http://www.memsic.com/userfiles/files/Datasheets/WSN/micaz_datasheet-t.pdf, 2015.

[61] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-level sensor network simulation with cooja," in *Local computer networks, proceedings 2006 31st IEEE conference on*, 2006, pp. 641-648.

[62] A. A. AL-Roubaiey, "Energy-Aware Publish/Subscribe DDS-Based Middleware for Wireless Sensor and Actuator Networks," *PhD Dissertation,* 2015.

[63] D. Riehle and H. Züllighoven, "Understanding and using patterns in software development," *TAPOS,* vol. 2, pp. 3-13, 1996.

[64] M. C. M. Thein and T. Thein, "An energy efficient cluster-head selection for wireless sensor networks," in *Intelligent systems, modelling and simulation (ISMS), 2010 international conference on*, 2010, pp. 287-291.

[65] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *System sciences, 2000. Proceedings of the 33rd annual Hawaii international conference on*, 2000, p. 10 pp. vol. 2.

Vitae

Awadh Moqbel Nasser Gaamel – M.S. in Computer Networks

PERSONAL Details

Date of Birth: 17/Feb/1987
Place of Birth: Najran-Saudi Arabia
Nationality: Yemeni
Gender: Male
Mobil: +966504838943
Personal Email: eng.awadh.moqbel@gmail.com
Web: www.researchgate.net/profile/Awadh_Gaamel

Education

2015-2017 **M.S in Computer Networks "Enhanced Energy Aware TinyDDS Publish/Subscribe Protocol (E-EATDDS)"**
King Fahd University of Petroleum and Minerals (KFUPM), Dhahran, Saudi Arabia
Supervisor: Prof. Tarek Sheltami, leading researcher in the area of ad hoc and sensor networks.

This study aims to develop and implement a new publish/ subscribe approach based on DDS (Data Distribution Service) standard. The proposed approach is called Enhanced Energy Aware TinyDDS (E-EATDDS). TinyOS simulator (TOSSIM) is used in this research along with different programming languages (C, nesC, java, python, awk). Also, Performance evaluation and extensive simulation tests were conducted on three TinyDDS approaches, namely Default TDDS, Broker-Less TDDS, and Hybrid TDDS in terms different metrics.

GPA: 3.679 out of 4

2008-2012 **B.S in Computer Engineering and Information Technology**
Sana'a University, Sana'a, Yemen

First Class (89.12%)
GPA: 3.56 out of 4

I got my B.S degree based on grant from Ministry of Higher education in Yemen. My graduation project was **"Tourist Regiment Management System for Tourism Agencies & Short Message Service (SMS) "**. In that project, I used Oracle, SQL commands and ActiveX Controls.

Honors and certifications

- 1st position of Bachelor degree in Computer Engineering and information technology.
- Cisco Certification CCNA1 with honor letters.
- Certificate of Certified Project Manager (C.P.M).
- Certificate of computer and maintenance

TECHNICAL AND PROGRAMS KNOWLEDGE

- Oracle SQL and developer
- C, C++, C#, nesC, Java
- TOSSIM Simulator
- OPNET simulator
- DDS Middleware
- TinyOS Sensors OS
- NCTUNs simulator
- AWK (scripting language)

LANGUAGES

- Arabic – native language
- English – speak and read/write

PUBLICATIONS

- **Gaamel, Awadh Moqbel**, Barakat Pravin Maratha, Tarek Rahil Sheltami, and Elhadi M. Shakshuki. "Fault-Tolerance Evaluation of VANET Under Different Data Dissemination Models." *International Journal of Vehicular Telematics and Infotainment Systems (IJVTIS)* 1, no. 1 (2017): 54-68. **(Published)**

- Shaheen, Ahmad, **Awadh Gaamel**, and Abdulqader Bahaj. "Comparison and analysis study between AODV and DSR routing protocols in vanet with IEEE 802.11 b." *J. Ubiquit. Syst. Pervasive Netw* 7, no. 1 (2016): 07-12. **(Published)**

- Talal Alkharobi**, Awadh Gaamel,** AHMED BINSAHAQ. (2017). An Efficient Cheating Identification Scheme in Secret Sharing. *International Journal of Information Security and Privacy (IJISP)*. **(Submitted)**

- **Awadh Gaamel**, Tarek Sheltami, Anas Al-Roubaiey, Elhadi Shakshuki. " Broker-Less Middleware for WSAN Performance Evaluation". *In the 12th International Conference on Future Networks and Communications (FNC-2017)*. **(Accepted)**

REFERENCES

Prof. Tarek Sheltami (supervisor)
Professor
Computer Engineering Department
King Fahd University of Petroleum & Minerals
P.O. Box 89, Dhahran 31261, Saudi Arabia
Phone: +966-3-860-4678
Fax: +966-3-860-3059
Email: tarek@kfupm.edu.sa
Homepage: http://faculty.kfupm.edu.sa/coe/tarek/

Dr. Talal Alkharobi
Assistant professor
Computer Engineering Department
King Fahd University of Petroleum & Minerals
P.O. Box 1498, Dhahran, 31261, Saudi Arabia
Phone: +966 3 860-2956
Fax: +966 3 860-2619
Email: talalkh@kfupm.edu.sa
Homepage: http://faculty.kfupm.edu.sa/COE/talalkh/

Dr. ANAS A. HASAN AL-ROUBAIEY
Assistant professor
Computer Engineering Department
King Fahd University of Petroleum & Minerals
P.O. Box 1498, Dhahran, 31261, Saudi Arabia
Phone: +966 3 860-1423
Fax: +966 3 860-3059
Academic Email: roubaiey@kfupm.edu.sa
Personal Email: saba717671@hotmail.com
Web: www.researchgate.net/profile/Anas_Al-Roubaiey/contributions?ev=prf_act